Data Engineering

Technologies

And how to become one…

Kian Sahafi

Introduction

Who is a data Engineer

- A data engineer is **an IT worker whose primary job is to prepare data for analytical or operational uses**. These software engineers are typically responsible for building data pipelines to bring together information from different source systems.
- Data engineering is one of the most popular and in-demand jobs among the big data domain across the world.

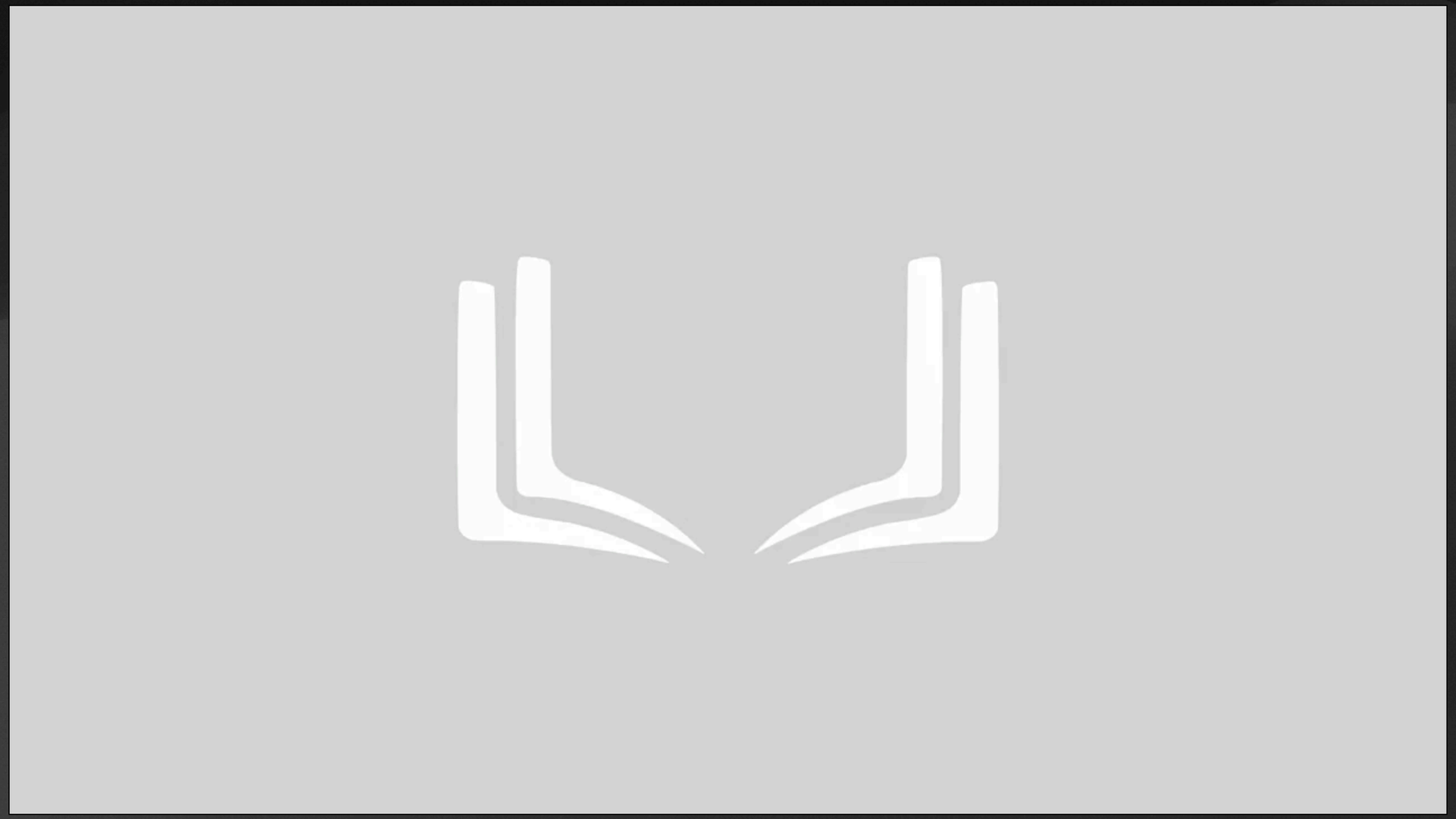
But what do they do?

- Data Engineers build monitor and refine complex data models to help organizations improve their business outcomes by harnessing data power.
- In other words they work in a variety of settings to build systems that collect, manage, and convert raw data into usable information for data scientists and business analysts to interpret.

And what are they trying to achieve?

- Their ultimate goal is to make data accessible so that organizations can use it to evaluate and optimize their performance.

But now lets see what a day in a life of an data engineer is like:



Ok, but What skill would you need to be a Data Engineer?

- **Coding:** Proficiency in coding languages is essential to this role, so consider taking courses to learn and practice your skills. Common programming languages include SQL, NoSQL, Python, Java, R, and Scala.

- **Relational and non-relational databases:** Databases rank among the most common solutions for data storage. You should be familiar with both relational and non-relational databases, and how they work.

- **ETL (extract, transform, and load) systems:** ETL is the process by which you'll move data from databases and other sources into a single repository, like a data warehouse. Common ETL tools include Xplenty, Stitch, Alooma, and Talend.

- **Data security:** While some companies might have dedicated data security teams, many data engineers are still tasked with securely managing and storing data to protect it from loss or theft.

- **Data storage:** Not all types of data should be stored the same way, especially when it comes to big data. As you design data solutions for a company, you'll want to know when to use a data lake versus a data warehouse, for example.

- **Automation and scripting:** Automation is a necessary part of working with big data simply because organizations are able to collect so much information. You should be able to write scripts to automate repetitive tasks.

- **Machine learning:** While machine learning is more the concern of data scientists, it can be helpful to have a grasp of the basic concepts to better understand the needs of data scientists on your team.

- **Big data tools:** Data engineers don't just work with regular data. They're often tasked with managing big data. Tools and technologies are evolving and vary by company, but some popular ones include Hadoop, MongoDB, Kafka and Spark.

- **Cloud computing:** You'll need to understand cloud storage and cloud computing as companies increasingly trade physical servers for cloud services. Beginners may consider a course in Amazon Web Services (AWS) or Google Cloud.

Who is AWS? (company)

Amazon Web Services (AWS) is a cloud computing platform offered by Amazon.com that provides a suite of services for building and running applications and websites. These services include computing, storage, database, analytics, machine learning, security, and many other functionalities, all of which can be accessed over the internet.

AWS was launched in 2002 and has since become one of the leading cloud computing platforms in the world. It provides a wide range of services to businesses, organizations, and individuals, enabling them to build and run their applications and websites on top of the AWS infrastructure.

AWS services are available on a pay-as-you-go basis, allowing customers to only pay for the resources they use. This makes it a flexible and cost-effective solution for businesses, as they can scale their resources up or down as needed without having to make significant upfront investments in hardware and infrastructure.

Why AWS is getting bold now days?

There are several reasons why AWS has become popular in recent years:

1. Scalability: AWS allows businesses to scale their resources up or down as needed, which makes it a flexible solution for companies that experience fluctuating workloads.
2. Cost-effectiveness: AWS charges customers on a pay-as-you-go basis, so they only pay for the resources they use. This makes it a cost-effective solution for businesses, as they don't have to make significant upfront investments in hardware and infrastructure.
3. Wide range of services: AWS offers a wide range of services, including computing, storage, database, analytics, machine learning, security, and many others. This allows businesses to build and run a variety of different applications and websites on top of the AWS platform.
4. Reliability: AWS has a strong track record of uptime and reliability, which is important for businesses that rely on the platform to run their applications and websites.
5. Global presence: AWS has a global infrastructure with regions and availability zones located around the world. This allows businesses to run their applications and websites in the region that is closest to their customers, which can improve performance and reduce latency.

Is AWS a market place?

Yes, AWS is a marketplace that allows businesses and individuals to buy and sell a wide range of cloud computing services. AWS provides a platform for vendors to offer their services, and customers can browse and purchase these services through the AWS website.

AWS offers a variety of services, including computing, storage, database, analytics, machine learning, security, and many others. Customers can use these services to build and run their applications and websites on top of the AWS infrastructure.

AWS also offers a number of tools and resources for vendors to use in developing and offering their services on the AWS marketplace. This includes the AWS Partner Network, which is a global community of consulting and technology partners that can help businesses build and sell their services on AWS.

We are going to talk about Spark in a few minutes but lets talk about AWS Glue first

- What is AWS Glue?

- AWS Glue is a serverless data integration service that makes it easy for analytics users to discover, prepare, move, and integrate data from multiple sources. You can use it for analytics, machine learning, and application development. It also includes additional productivity and data ops tooling for authoring, running jobs, and implementing business workflows.

- With AWS Glue, you can discover and connect to more than 70 diverse data sources and manage your data in a centralized data catalog. You can visually create, run and monitor extract, transform, and load (ETL) pipelines to load data into your data lakes. Also, you can immediately search and query cataloged data using Amazon Athena, Amazon EMR, and Amazon Redshift Spectrum.

But how does It work?

Here's how AWS Glue works:

1. Data extraction: AWS Glue can extract data from a variety of sources, including Amazon S3, Amazon RDS, Amazon Redshift, and other data stores.
2. Data transformation: AWS Glue can transform the extracted data using a variety of transformations, such as filtering, sorting, and aggregating data.
3. Data loading: AWS Glue can load the transformed data into a variety of data stores, including Amazon S3, Amazon RDS, Amazon Redshift, and other data stores.

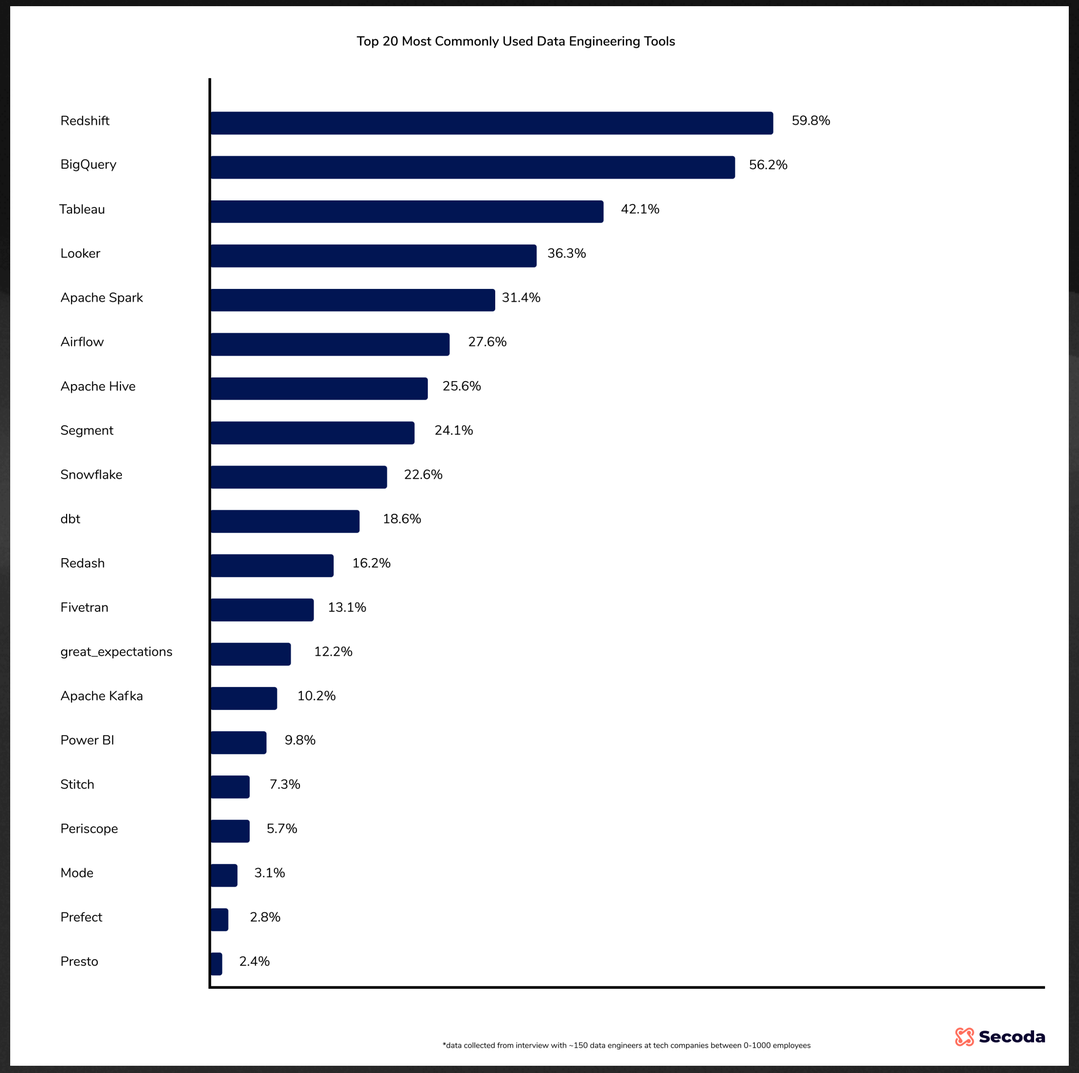
AWS Glue also includes a number of features to help users build and maintain their ETL jobs, including a visual development environment, a library of pre-built connectors and transformations, and the ability to schedule ETL jobs.

Data professionals talk about how they define data engineering and how it differs from data analytics and data science.

Let’s talk about the technologies now:

- These are the Top 20 Most Commonly Used Data Engineering Tools in the Year 2022

- Of course we cannot talk about all of them at the moment but we will try our best explain some of them.



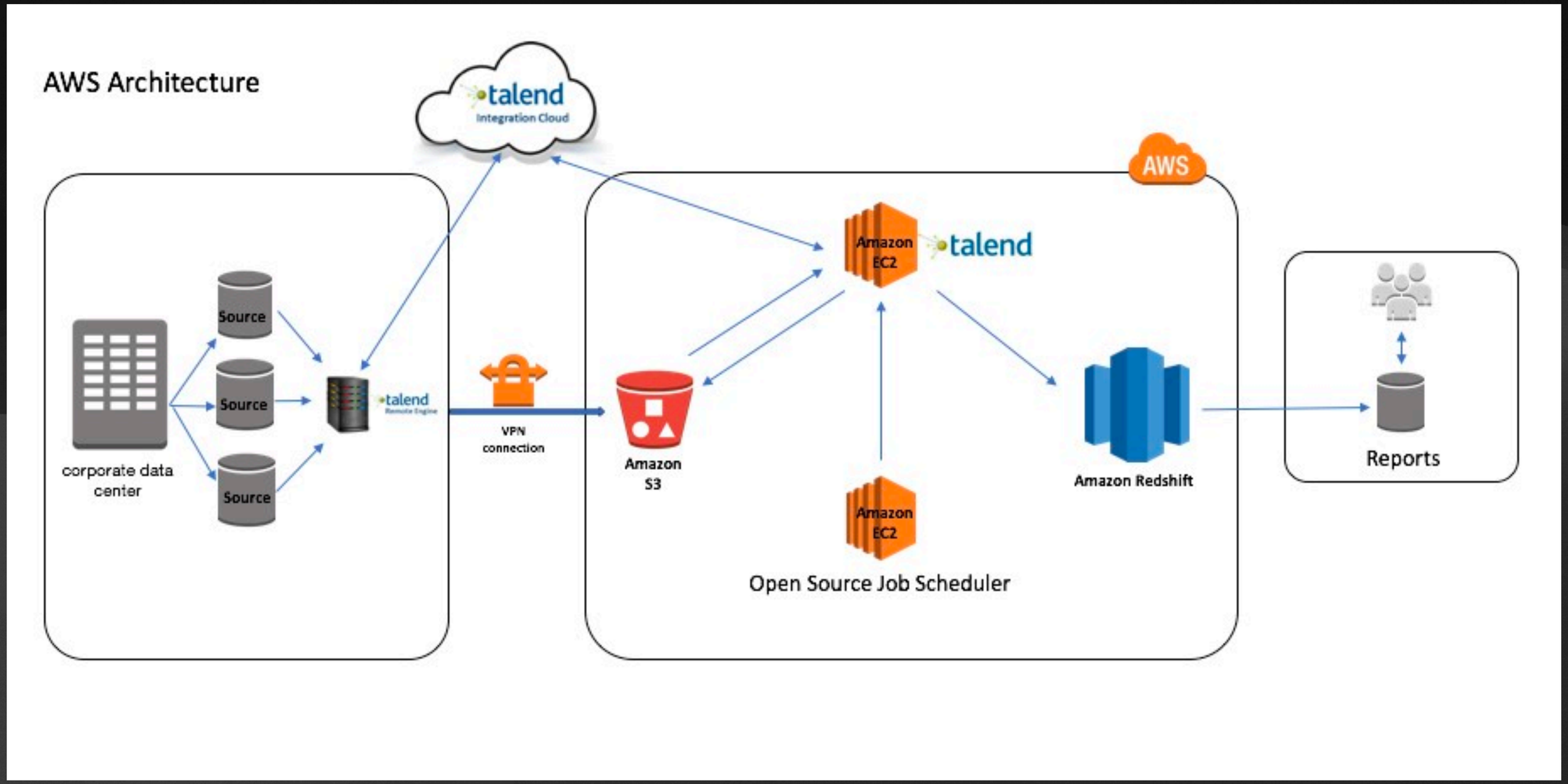
1. Amazon Redshift

Amazon Redshift is a fully-managed data warehouse service offered by Amazon Web Services (AWS). It is designed to handle large amounts of data and allows users to analyze data using SQL and business intelligence (BI) tools.

Amazon Redshift is based on a columnar data storage model, which allows it to efficiently store and retrieve data for fast querying and analysis. It also includes a variety of features to optimize performance, such as data compression and the ability to parallelize queries across multiple nodes.

Customers can use Amazon Redshift to store and analyze data for a wide range of applications, including business intelligence, data warehousing, analytics, and more. It is a cost-effective solution, as customers only pay for the resources they use and can scale their resources up or down as needed.

Using Amazon Redshift for Fast Analytical Reports



Result of Using AmazonRedshift

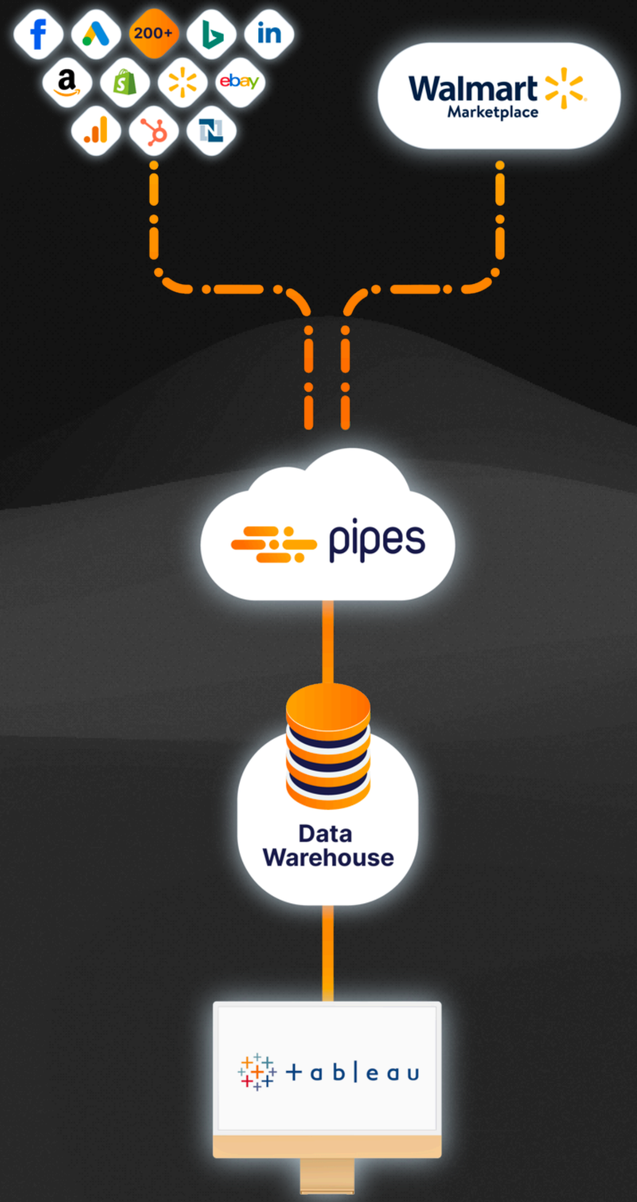
Table	Load time (sec)			End-to-end time (sec)		
	On-premises	Amazon Redshift	Improvement	On-premises	Amazon Redshift	Improvement
Table 1	617	129	5x	732	320	2x
Table 2	1766	184	9x	1767	305	6x
Table 3	308	63	5x	309	130	2x
Table 4	154	102	1.5x	2115	2126	0x

3. Tableau

- Tableau is an excellent data visualization and business intelligence tool used for reporting and analyzing vast volumes of data. It is an American company that started in 2003—in June 2019, Salesforce acquired Tableau. It helps users create different charts, graphs, maps, dashboards, and stories for visualizing and analyzing data, to help in making business decisions.
- It has some features like:
- Tableau supports powerful data discovery and exploration that enables users to answer important questions in seconds
- users without relevant experience can start immediately with creating visualizations using Tableau
- It can connect to several data sources that other BI tools do not support. Tableau enables users to create reports by joining and blending different datasets
- Tableau Server supports a centralized location to manage all published data sources within an organization

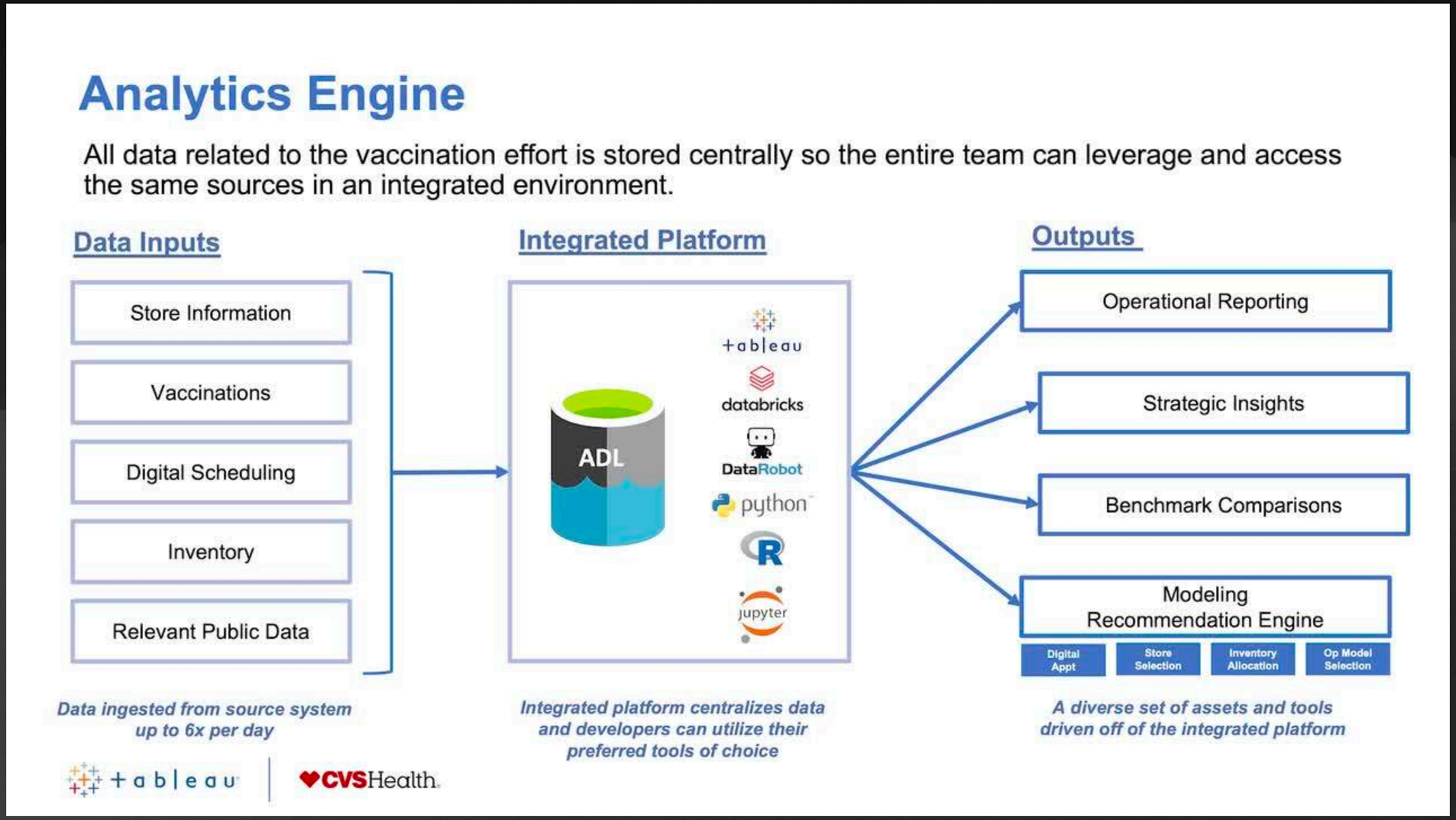
Usage of Tableau in Walmart

And how to connect your own data warehouse to Tableau



- This is how to get data from Walmart marketplace(and from other sources) into Tableau by locating it into your data warehouse that is connect to Tableau.
- Load your Walmart Marketplace data into your central data warehouse to analyze it with Tableau.

Tableau Usage in CVS Health Corporation of USA

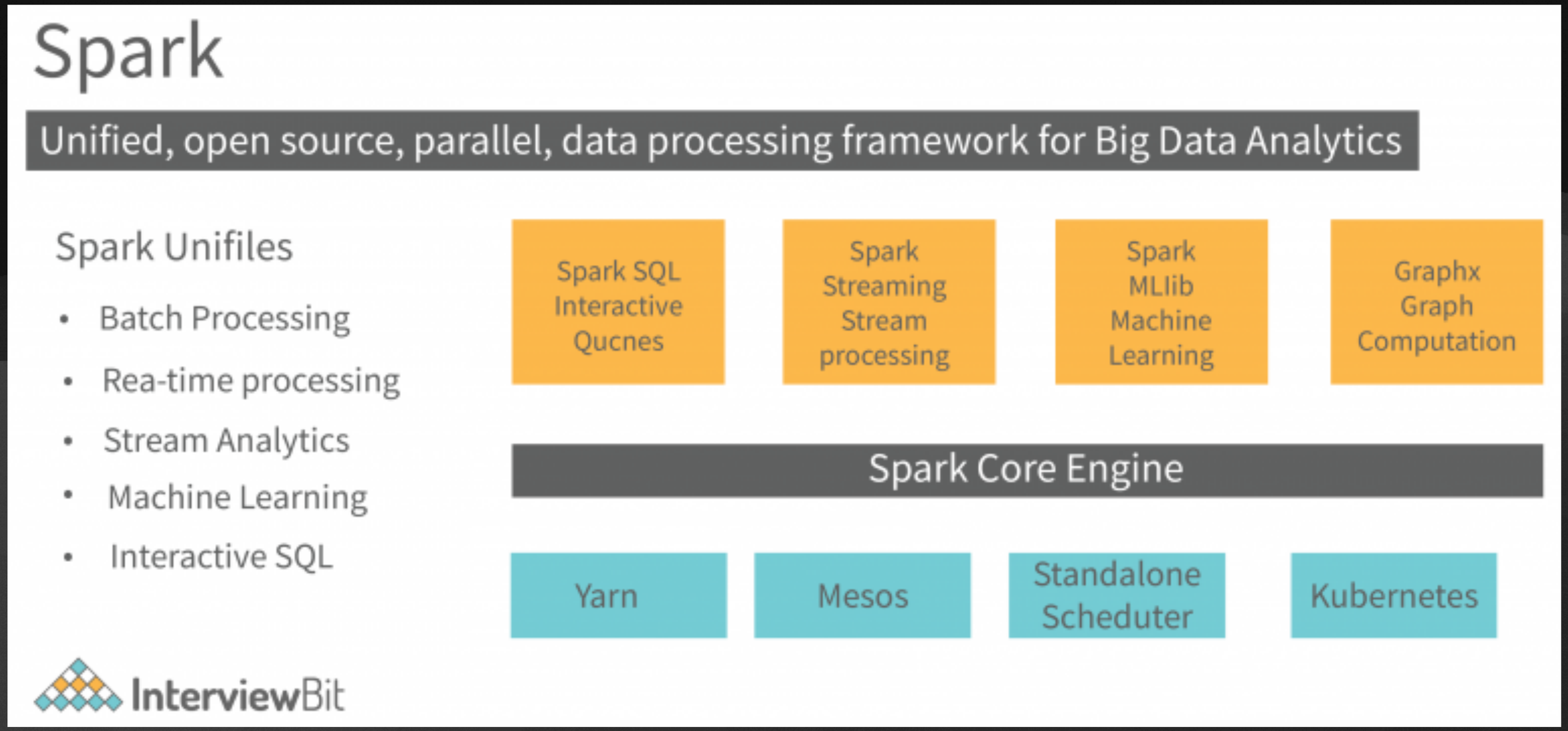


Spark on AWS

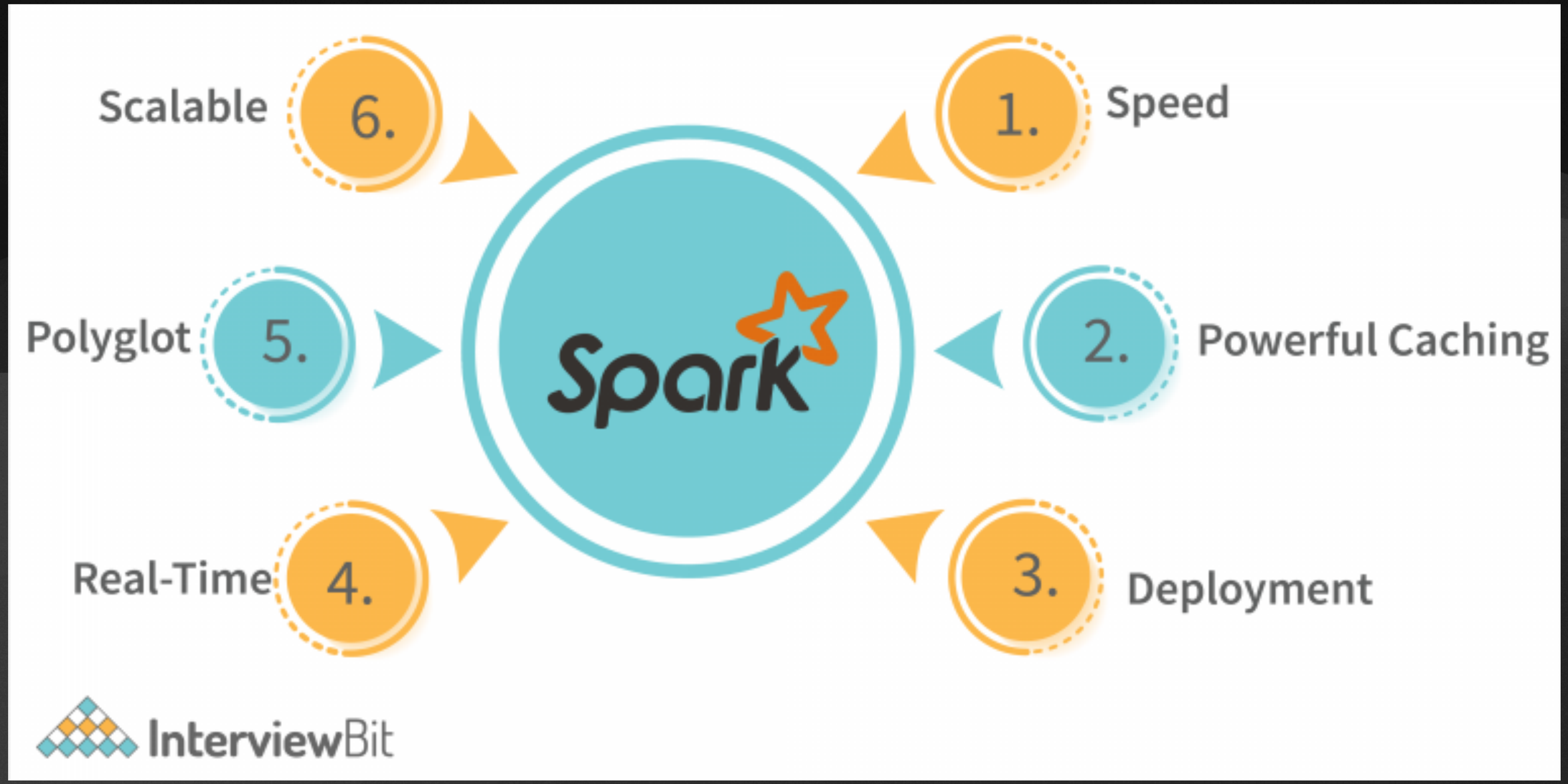
Sharing experience of
Using Spark on AWS infrastructure

Farbod Ahmadian

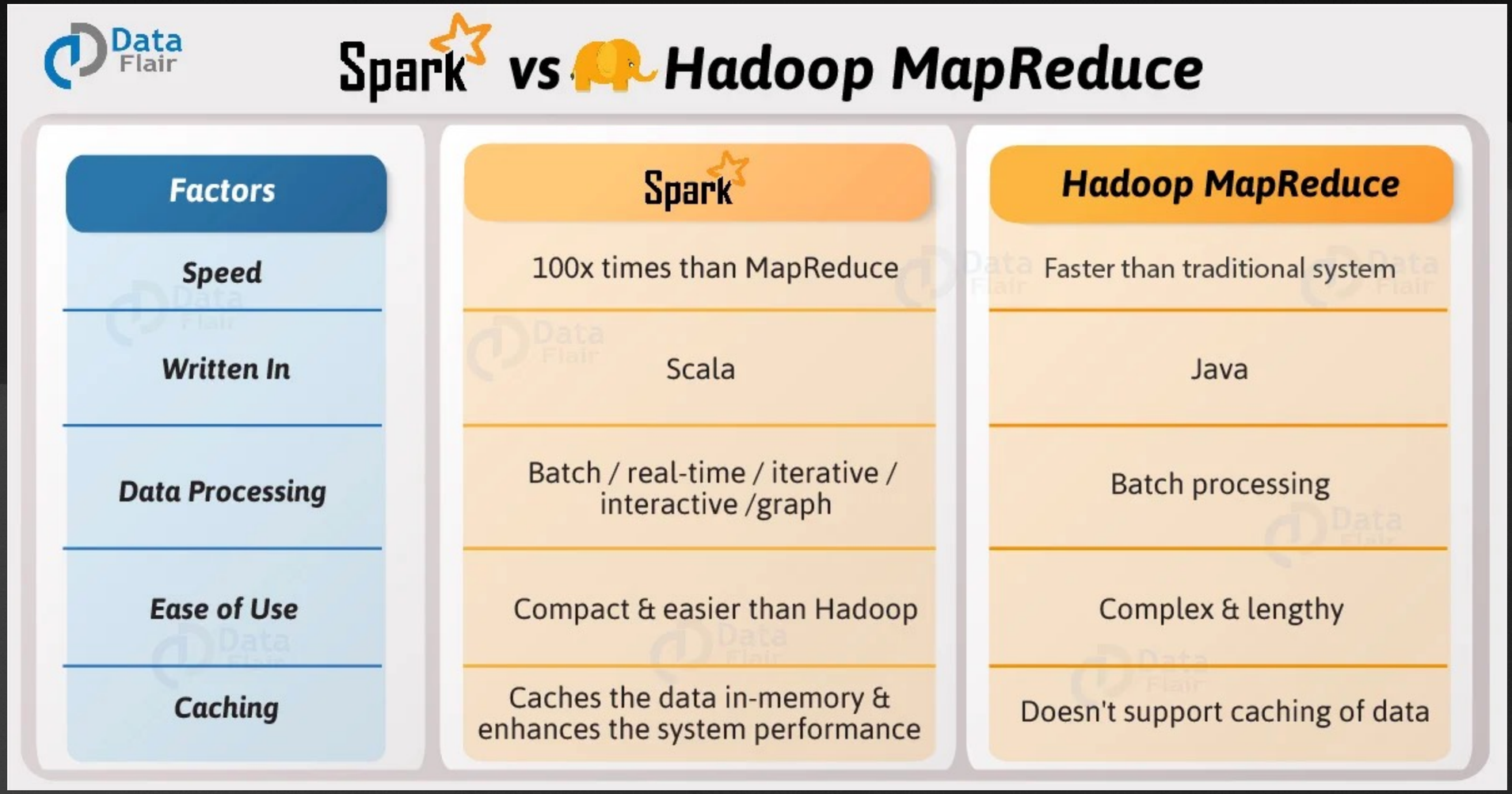
What is Spark?



Spark Features



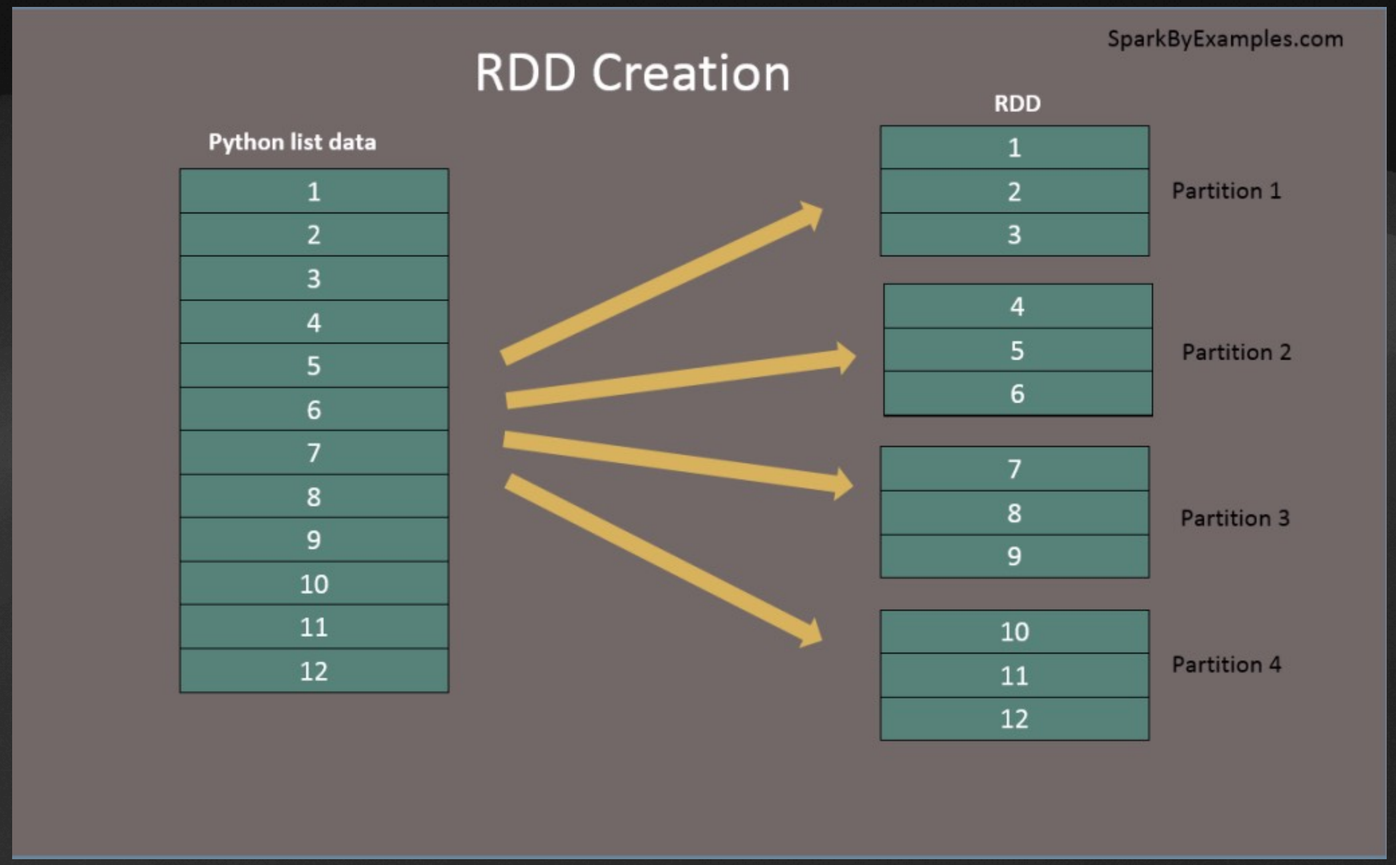
Apache Spark VS Hadoop MapReduce



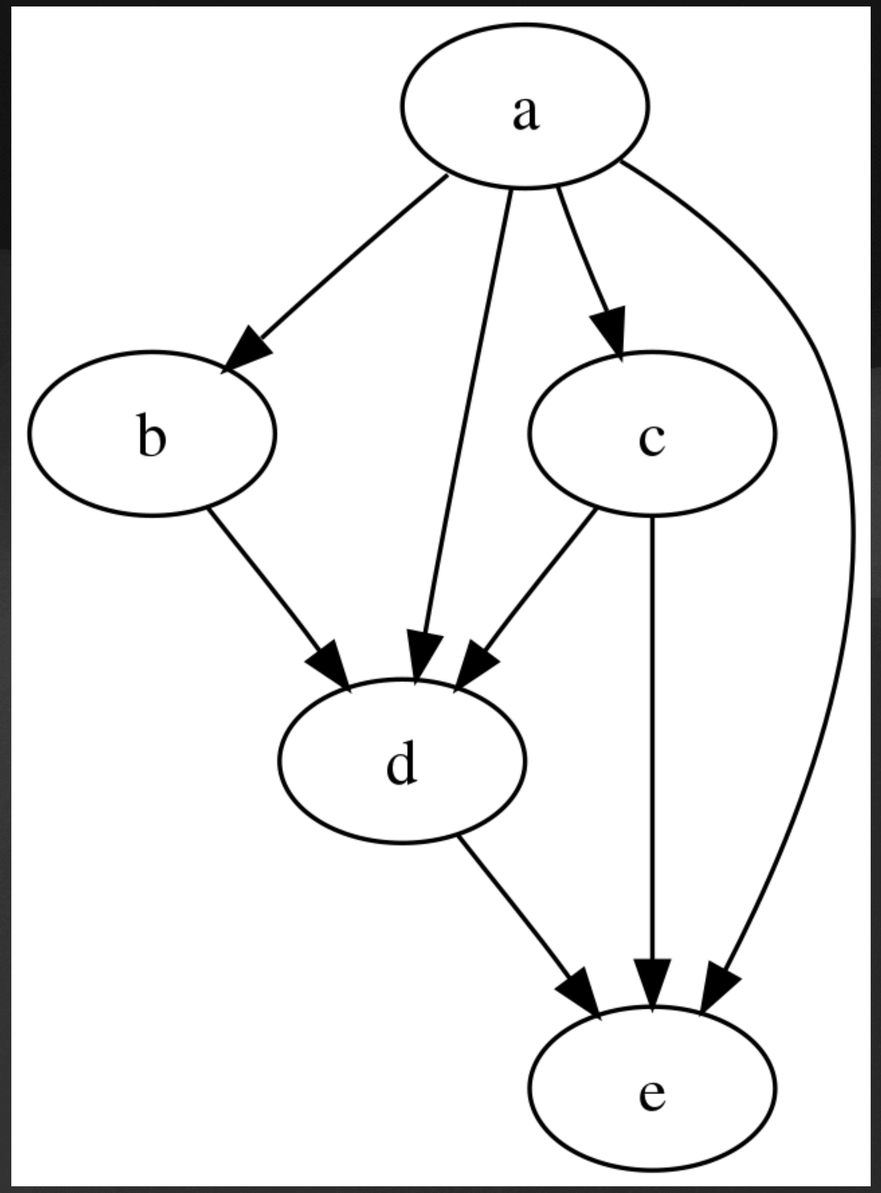
Factors	Spark	Hadoop MapReduce
Speed	100x times than MapReduce	Faster than traditional system
Written In	Scala	Java
Data Processing	Batch / real-time / iterative / interactive /graph	Batch processing
Ease of Use	Compact & easier than Hadoop	Complex & lengthy
Caching	Caches the data in-memory & enhances the system performance	Doesn't support caching of data

Two Main Abstractions of Apache Spark

- **Resilient Distributed Datasets (RDD):**
is a fundamental data structure of Spark and it is the primary data abstraction in Apache Spark and the Spark Core. RDDs are fault-tolerant, immutable distributed collections of objects, which means once you create an RDD you cannot change it. Each dataset in RDD is divided into
logical partitions, which can be computed on different nodes of the cluster.

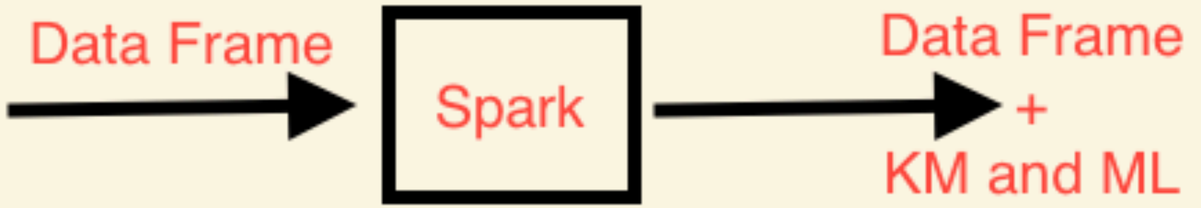


Directed Acyclic Graph (DAG)

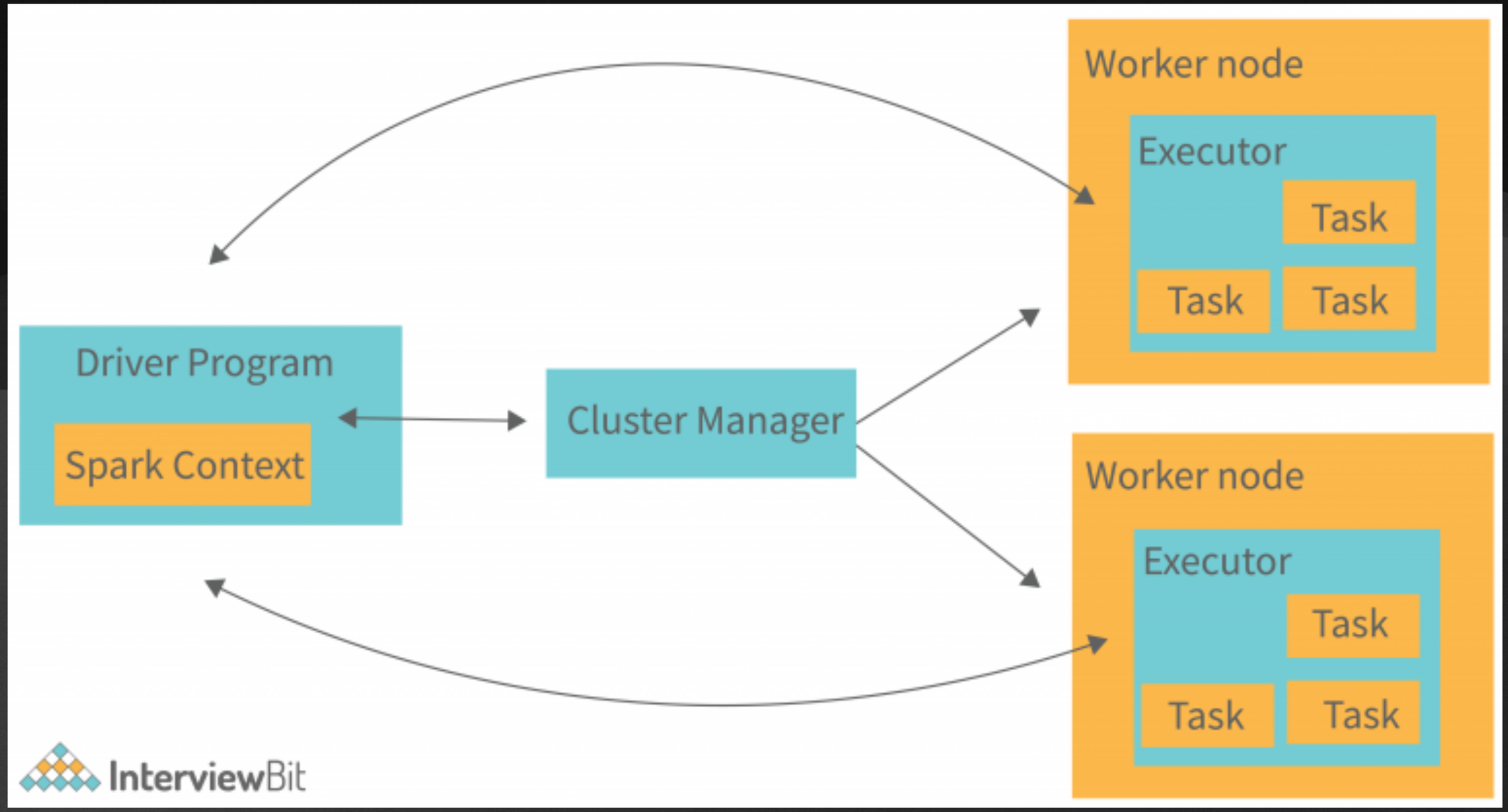


Spark Simple Transformation

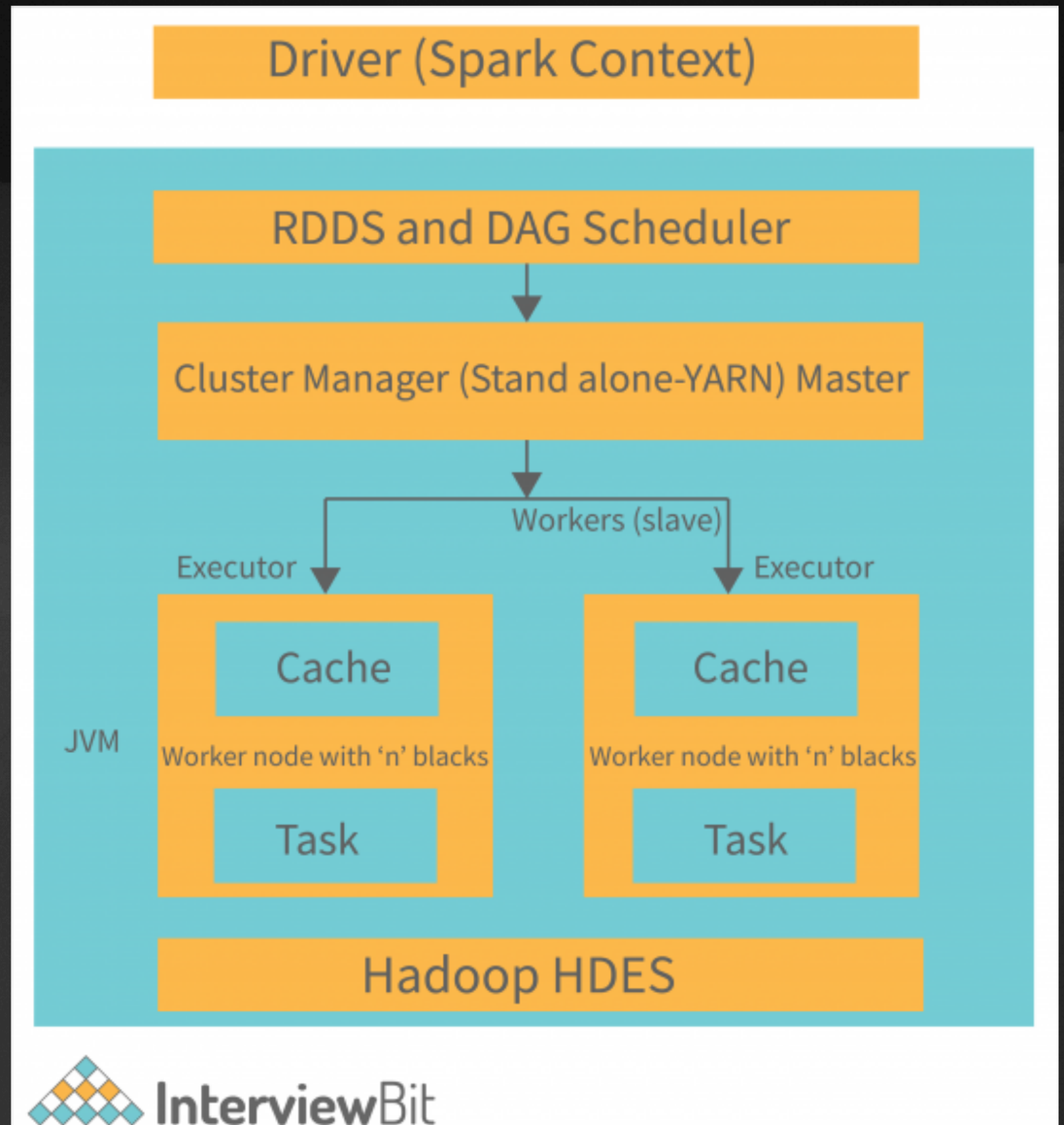
```scala
/** Adds miles and km column to DataFrame
  * @param source MileageColTuple containing construct name and amount
  * @return DataFrame with two added columns containing miles and km
  */
def addMileAndKilometerTransformer(
  source: MileageColTuple
): DataFrame => DataFrame = { df =>
  df.addColToStruct(
    s"${source.name}.amount_km",
    when(col(source.Unit) <=> "km", col(source.Value).cast(DoubleType))
      .otherwise(round(col(source.Value) * 1.60934, 2))      Bram Elfrink, 19 months ago
  ).addColToStruct(
    s"${source.name}.amount_miles",
    when(col(source.Unit) <=> "miles", col(source.Value).cast(DoubleType))
      .otherwise(round(col(source.Value) / 1.60934, 2))
  )
```



Spark Architecture

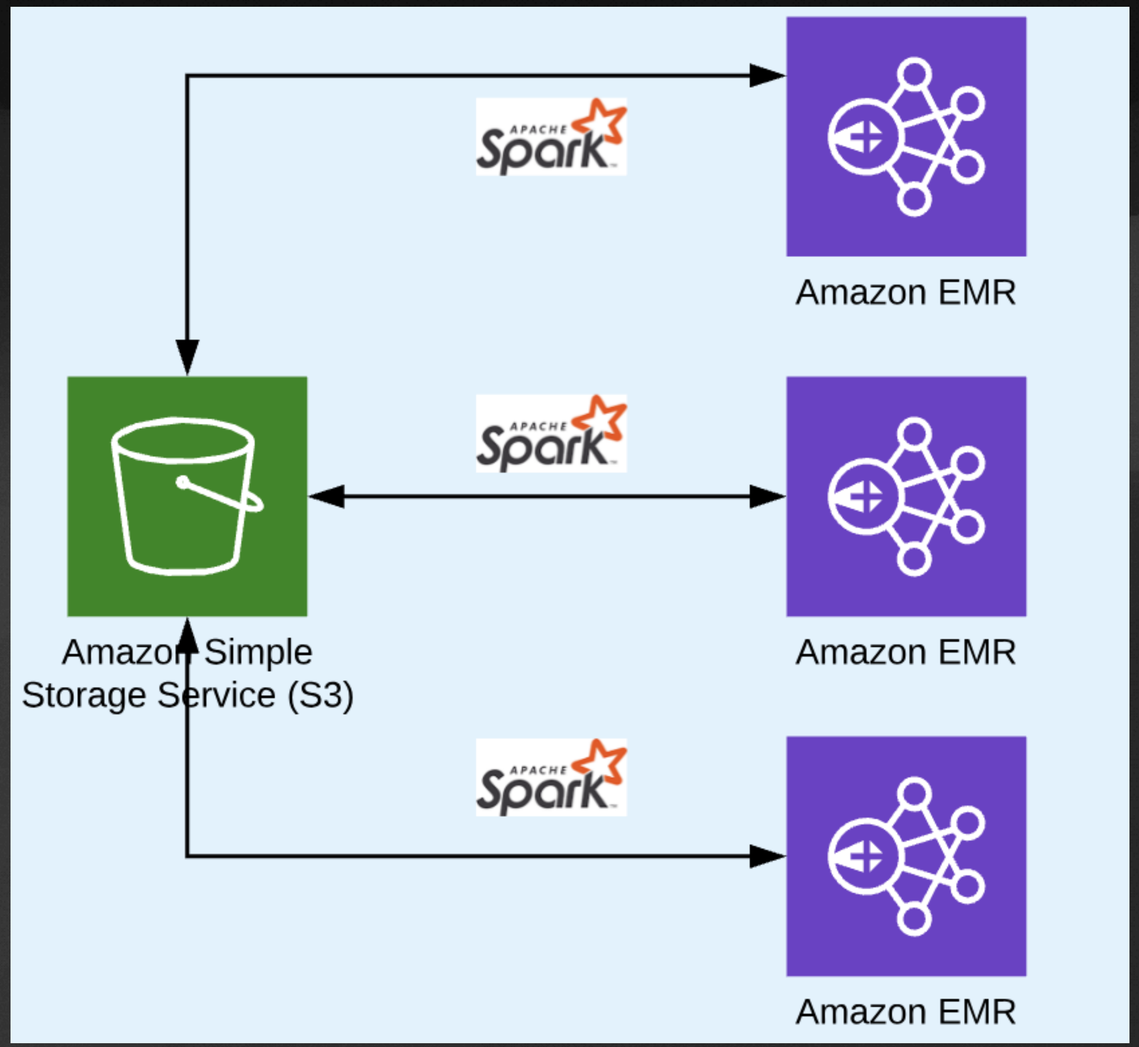


More Architecture!



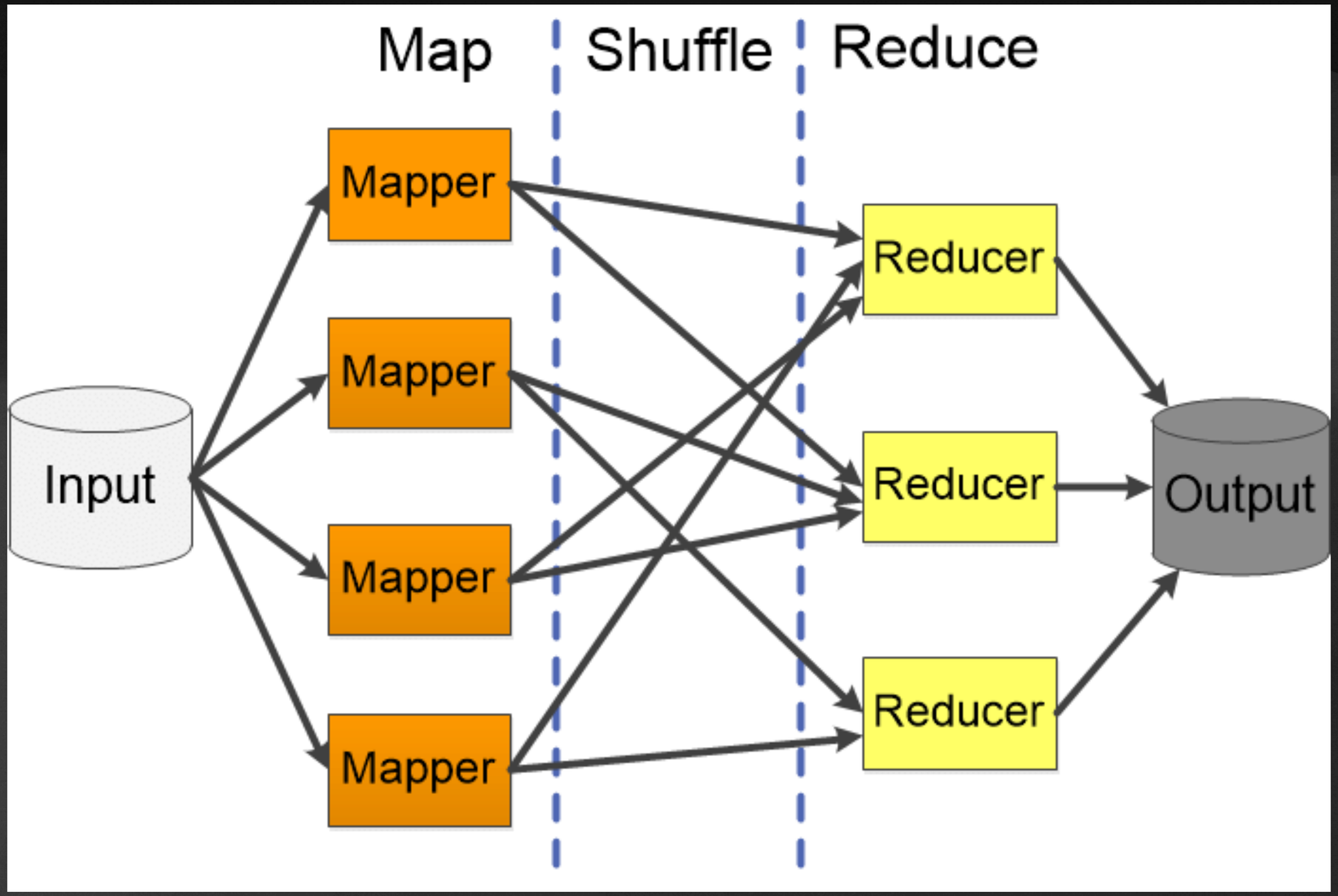
Infra? AWS comes to the game

Every application needs an Infrastructure to run on

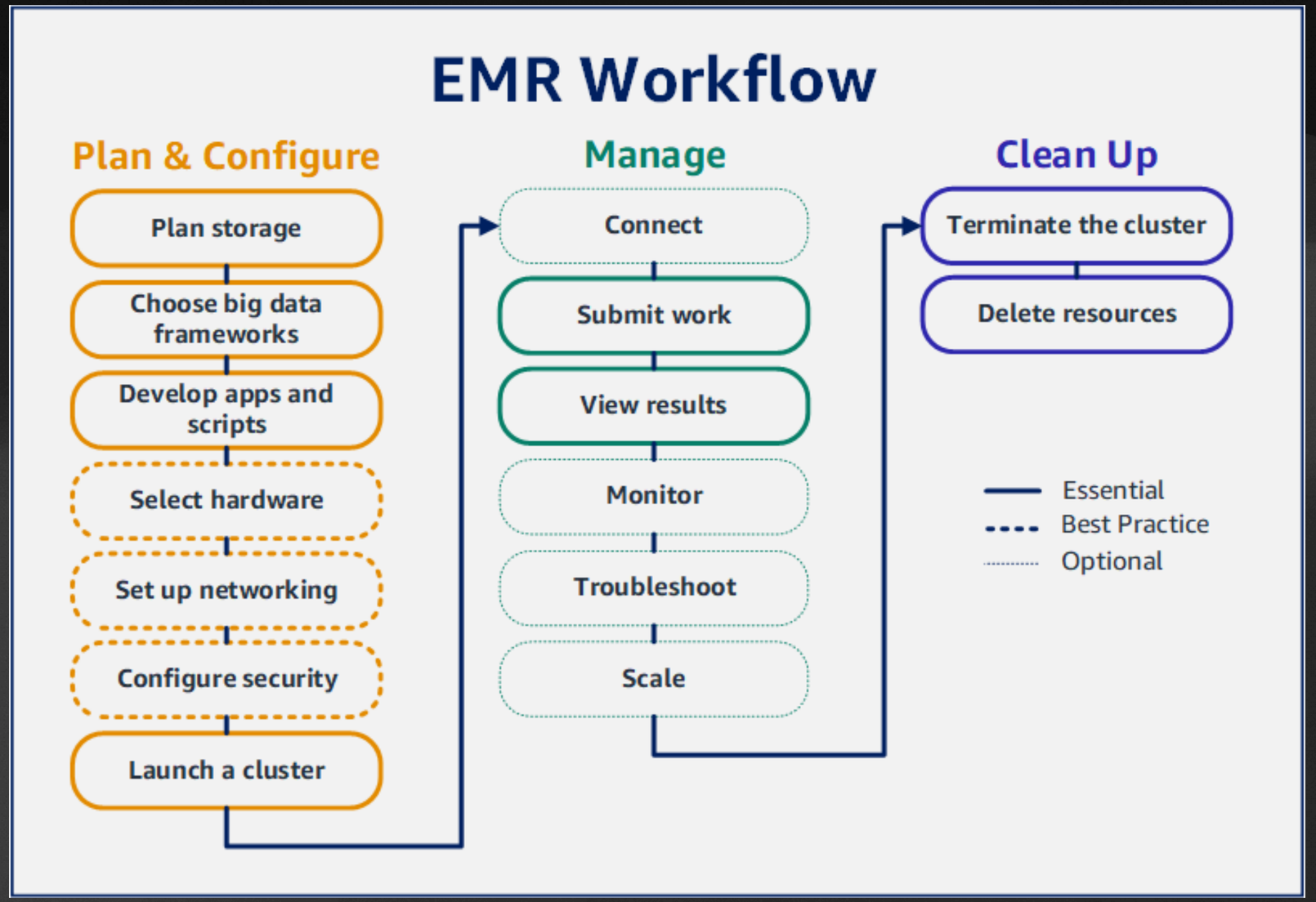


MapReduce Algorithm

Break task into smaller sub-tasks, **Map** each sub-task to a tread, **Reduce** in master node



Back to AWS: Elastic Map Reduce (EMR)

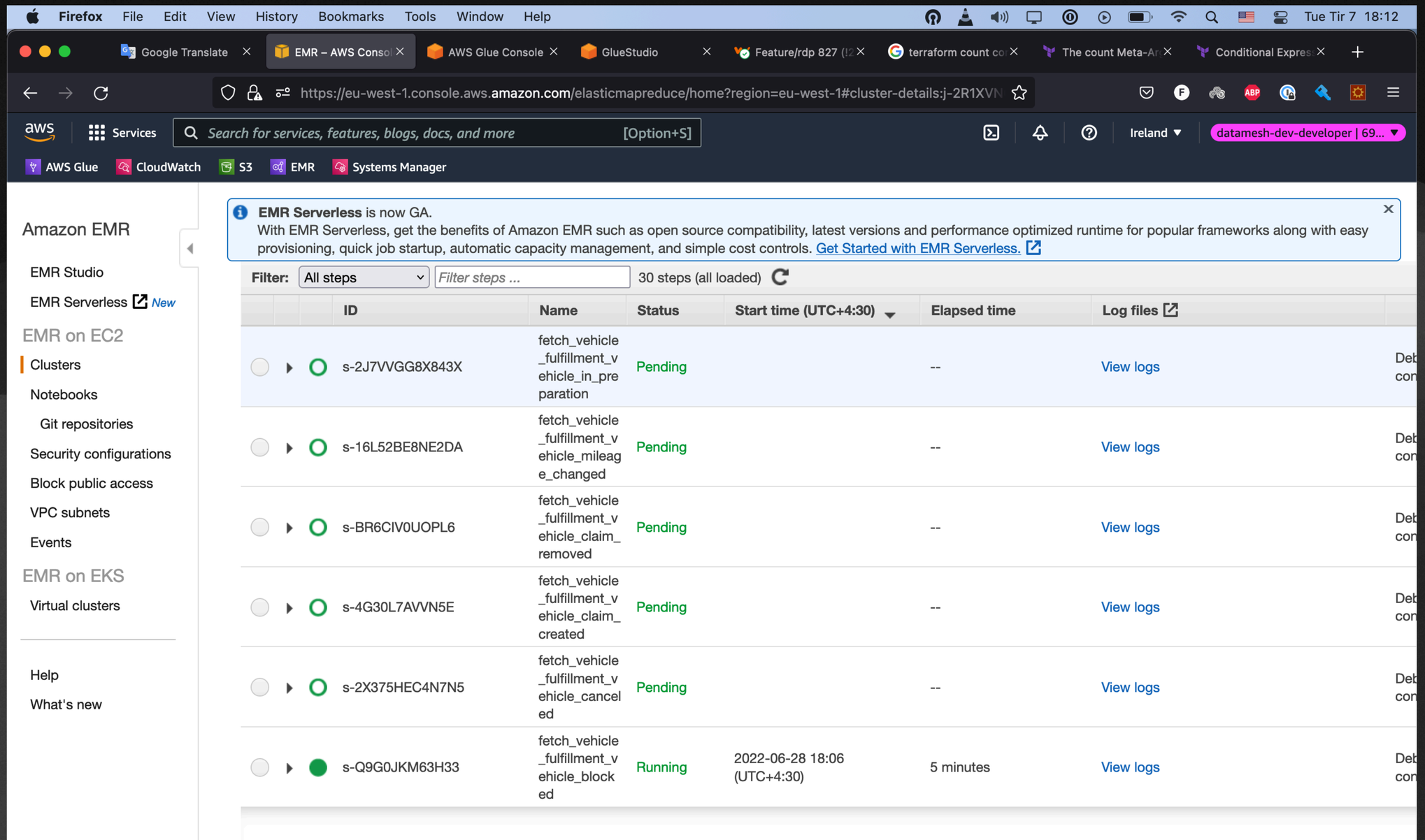


EMR Cost

Region: Europe (Ireland)

	Amazon EC2 Price (On Demand)	Amazon EMR Price
General Purpose - Current Generation		
m6a.xlarge	$0.1926 per hour	$0.0432 per hour
m6a.2xlarge	$0.3852 per hour	$0.0864 per hour
m6a.4xlarge	$0.7704 per hour	$0.1728 per hour
m6a.8xlarge	$1.5408 per hour	$0.3456 per hour
m6a.12xlarge	$2.3112 per hour	$0.5184 per hour
m6a.16xlarge	$3.0816 per hour	$0.6912 per hour
m6a.24xlarge	$4.6224 per hour	$1.0368 per hour
m6a.32xlarge	$6.1632 per hour	$1.3824 per hour
m6a.48xlarge	$9.2448 per hour	$2.0736 per hour
m6g.xlarge	$0.172 per hour	$0.039 per hour
m6g.2xlarge	$0.344 per hour	$0.077 per hour
m6g.4xlarge	$0.688 per hour	$0.154 per hour
m6g.8xlarge	$1.376 per hour	$0.308 per hour
m6g.12xlarge	$2.064 per hour	$0.462 per hour
m6g.16xlarge	$2.752 per hour	$0.616 per hour
m6i.xlarge	$0.214 per hour	$0.048 per hour
m6i.2xlarge	$0.428 per hour	$0.096 per hour

EMR Sample Console



Sample Code For Creating Task On EMR

```
try:
    response = emr_client.add_job_flow_steps(
        JobFlowId=cluster_id,
        Steps=[{
            'Name': name,
            'ActionOnFailure': 'CONTINUE',
            'HadoopJarStep': {
                'Jar': 'command-runner.jar',
                'Args': ['spark-submit', '--deploy-mode', 'cluster',
                         script_uri, *script_args]
            }
        }])
    step_id = response['StepIds'][0]
    logger.info("Started step with ID %s", step_id)
except ClientError:
    logger.exception("Couldn't start step %s with URI %s.", name, script_uri)
    raise
else:
    return step_id
```

Thanks For listening!

If you have any question now is the time ;-)

Presented by Kian Sahafi