GENERAL ASSEMBLY

# DATA SCIENCE

## SYD DAT 6

Week 6 – Cloud computing, Big Data & Spark
Wednesday 16th November

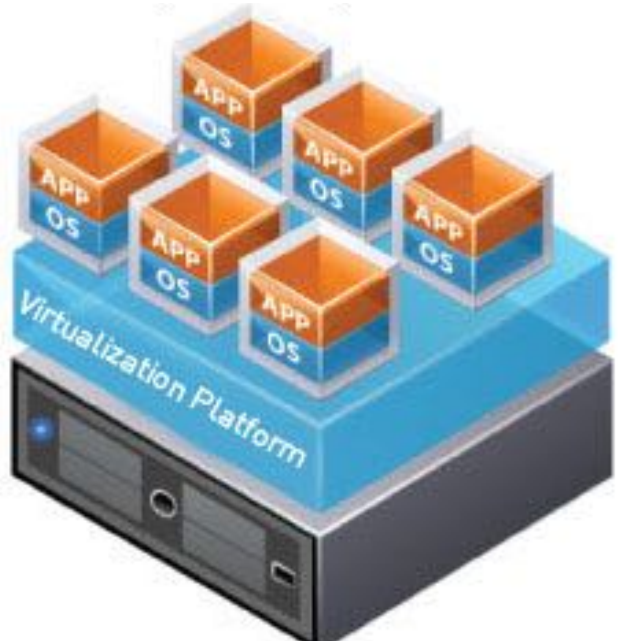
# AGENDA

1. Cloud Computing
2. Data Stores and Computation
3. Big Data
4. Spark
5. Lab
6. Real World Problem
7. Review

## DATA SCIENCE PART TIME COURSE

# CLOUD

# CLOUD


APP
OS
APP
OS
APP
OS
APP
OS
APP
OS
APP
OS
Virtualization Platform

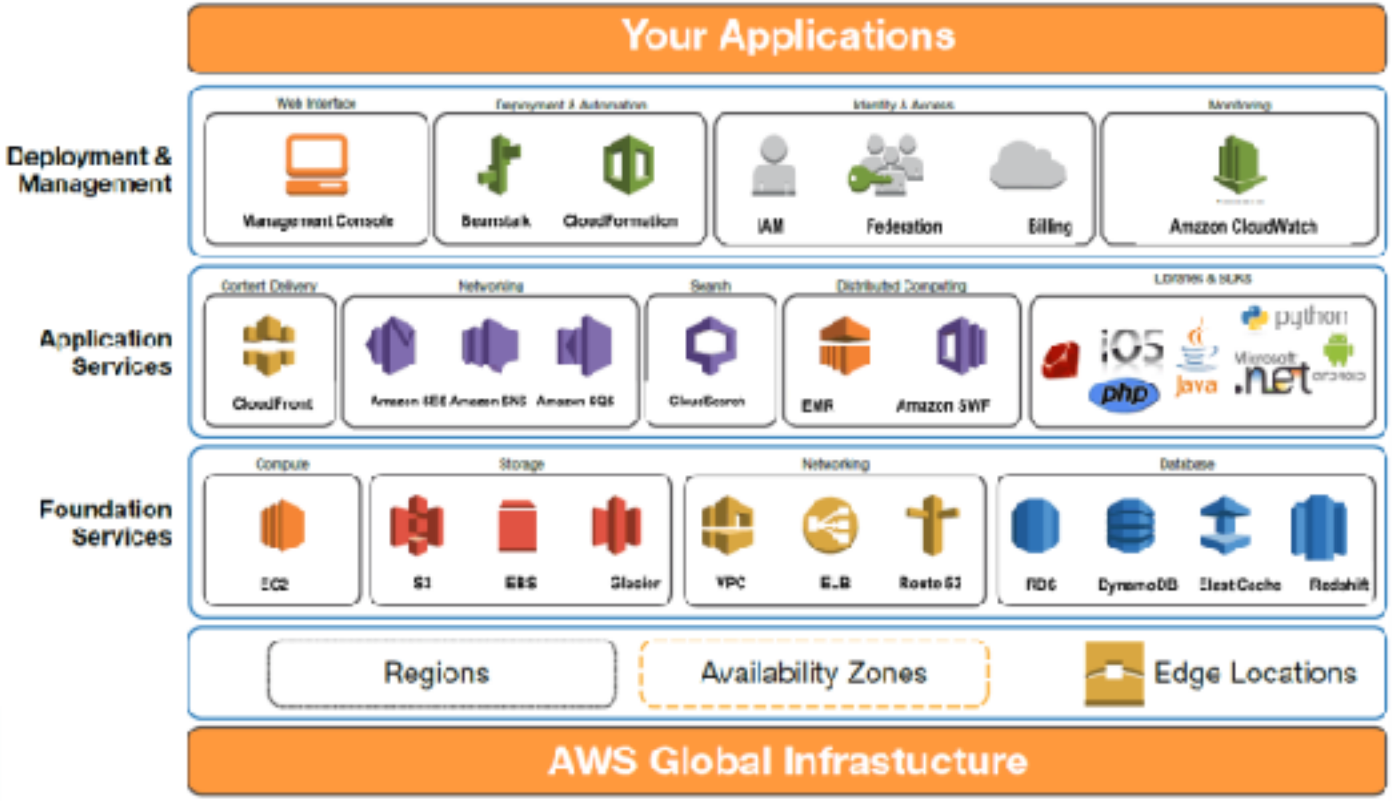
# PUBLIC CLOUD PROVIDERS

amazon
web services

Microsoft
Azure

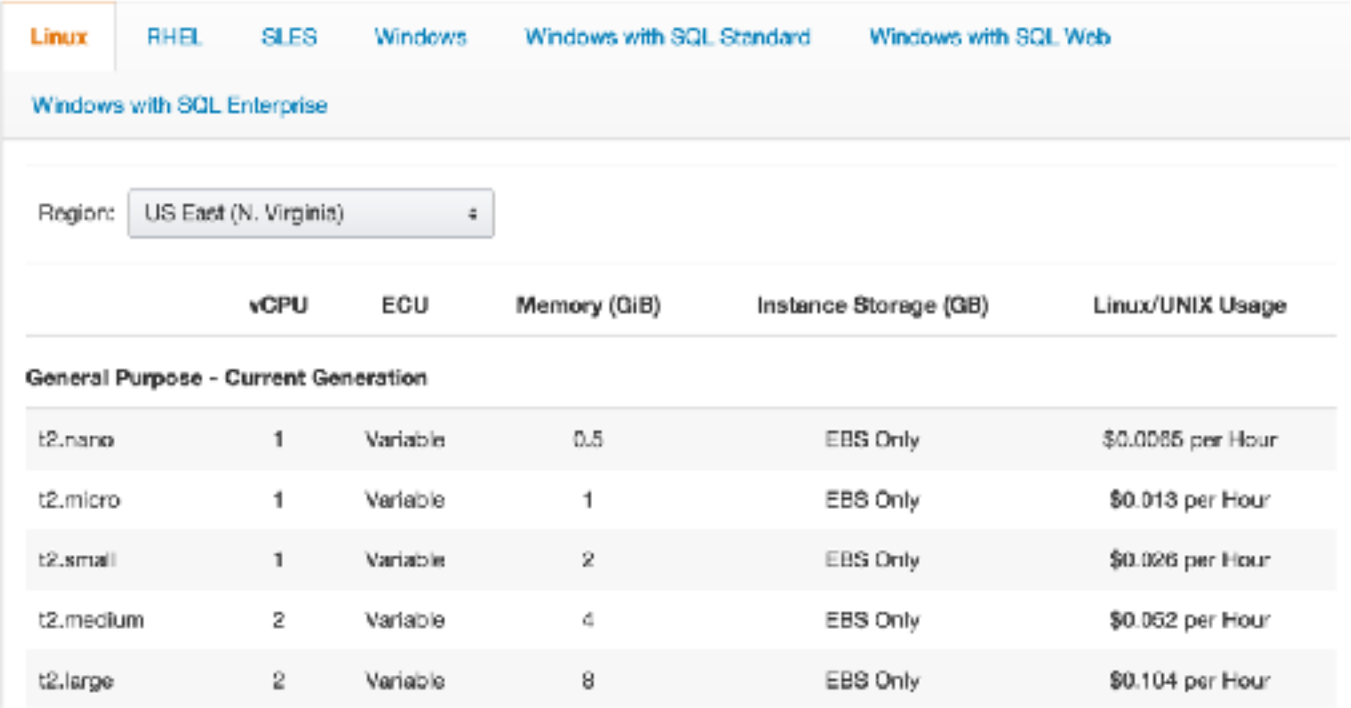
Google Cloud Platform

# AWS – WHAT IS IT AND WHY USE IT?



# AWS PRICES

https://aws.amazon.com/ec2/pricing/on-demand/

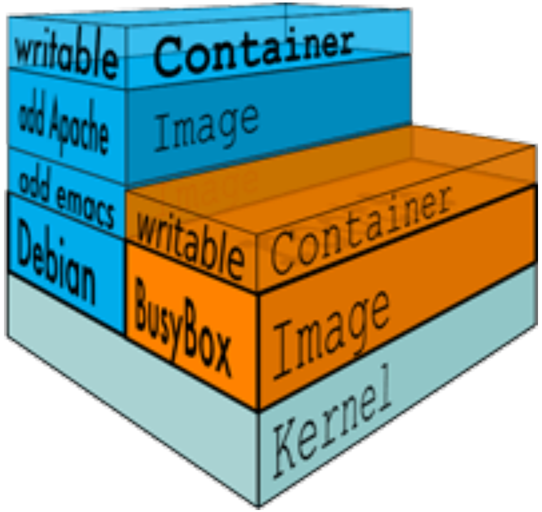


Region: US East (N. Virginia)

| | vCPU | ECU | Memory (GiB) | Instance Storage (GB) | Linux/UNIX Usage |
|---|---|---|---|---|---|
| **General Purpose - Current Generation** | | | | | |
| t2.nano | 1 | Variable | 0.5 | EBS Only | $0.0065 per Hour |
| t2.micro | 1 | Variable | 1 | EBS Only | $0.013 per Hour |
| t2.small | 1 | Variable | 2 | EBS Only | $0.026 per Hour |
| t2.medium | 2 | Variable | 4 | EBS Only | $0.052 per Hour |
| t2.large | 2 | Variable | 8 | EBS Only | $0.104 per Hour |

# DOCKER – WHAT IS IT?

Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server. This guarantees that it will always run the same, regardless of the environment it is running in.



- Lightweight
- Open
- Secure

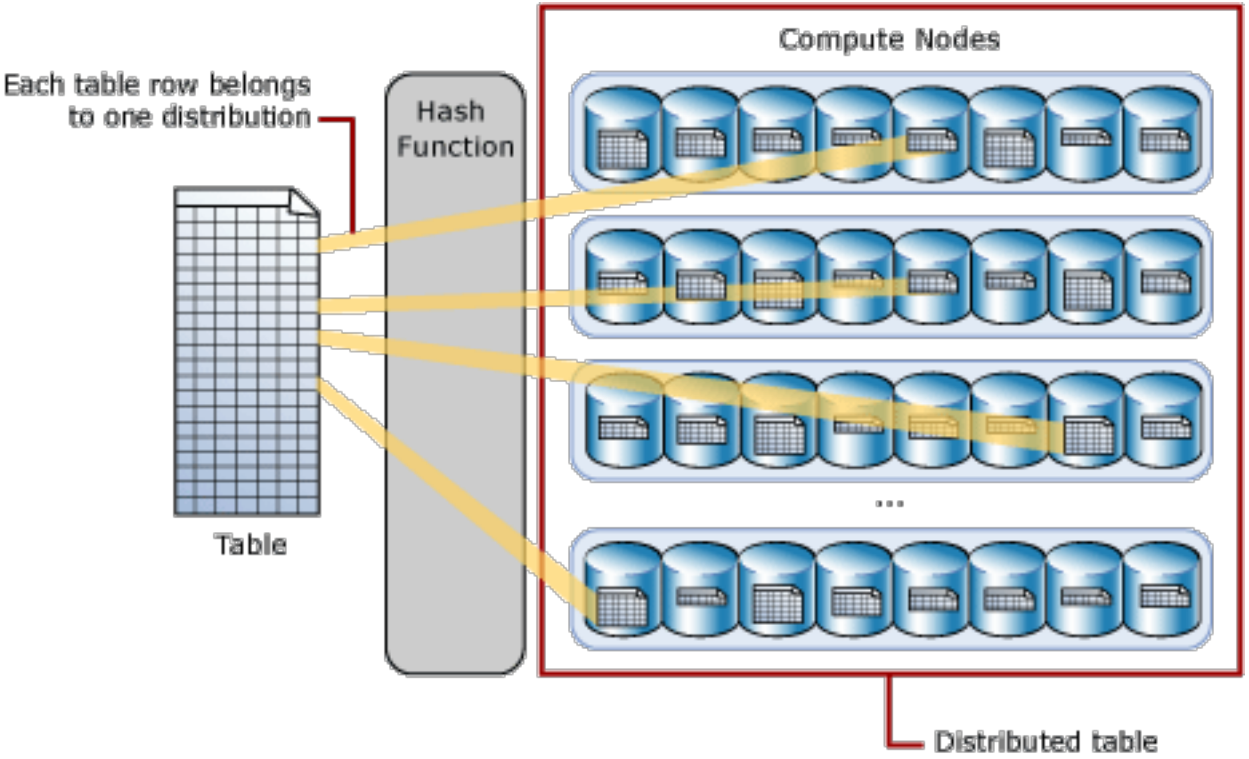
DATA SCIENCE PART TIME COURSE

# DATA STORES AND COMPUTATION

# RDBMS

Microsoft SQL Server
ORACLE
PostgreSQL
MySQL
SYBASE
IBM DB2

# MPP RDBMS - Massively Parallel Processing



TERADATA

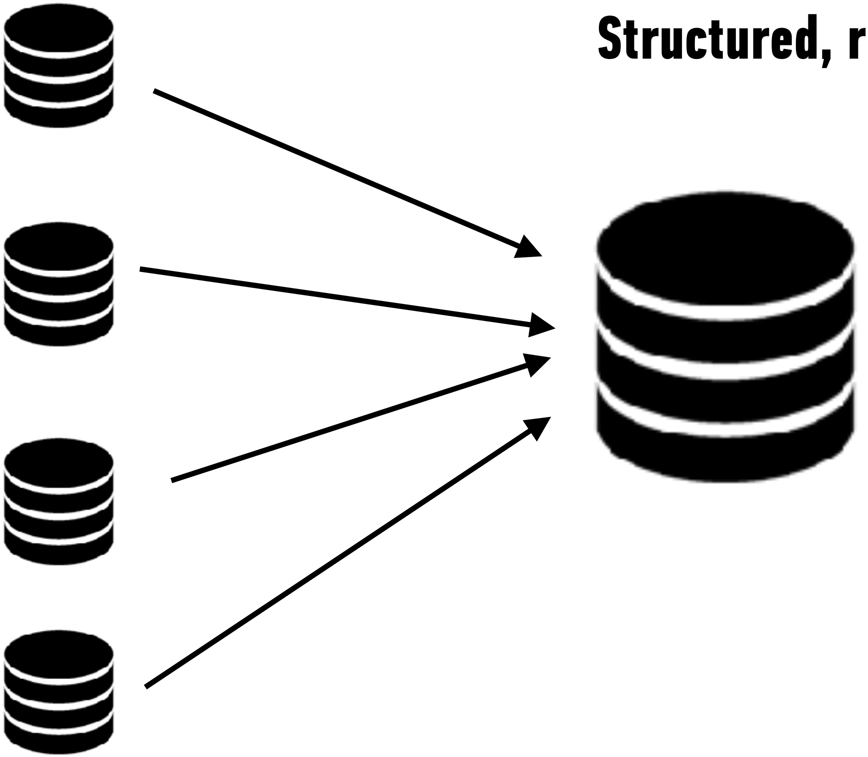
Greenplum

VERTICA
An HP Company

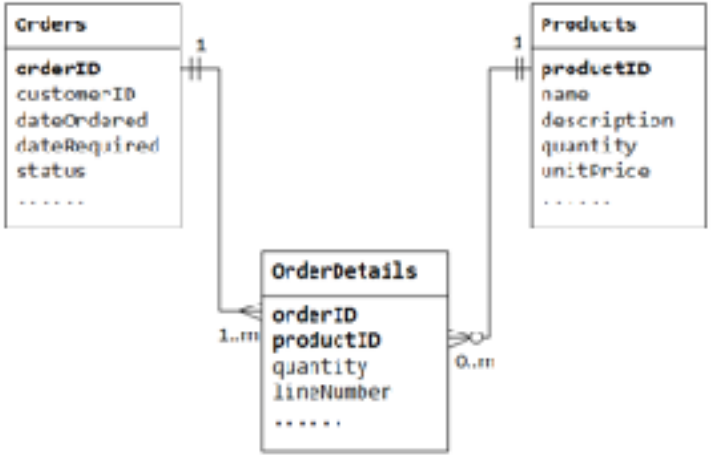
AMAZON REDSHIFT

# TRADITIONAL ARCHITECTURE

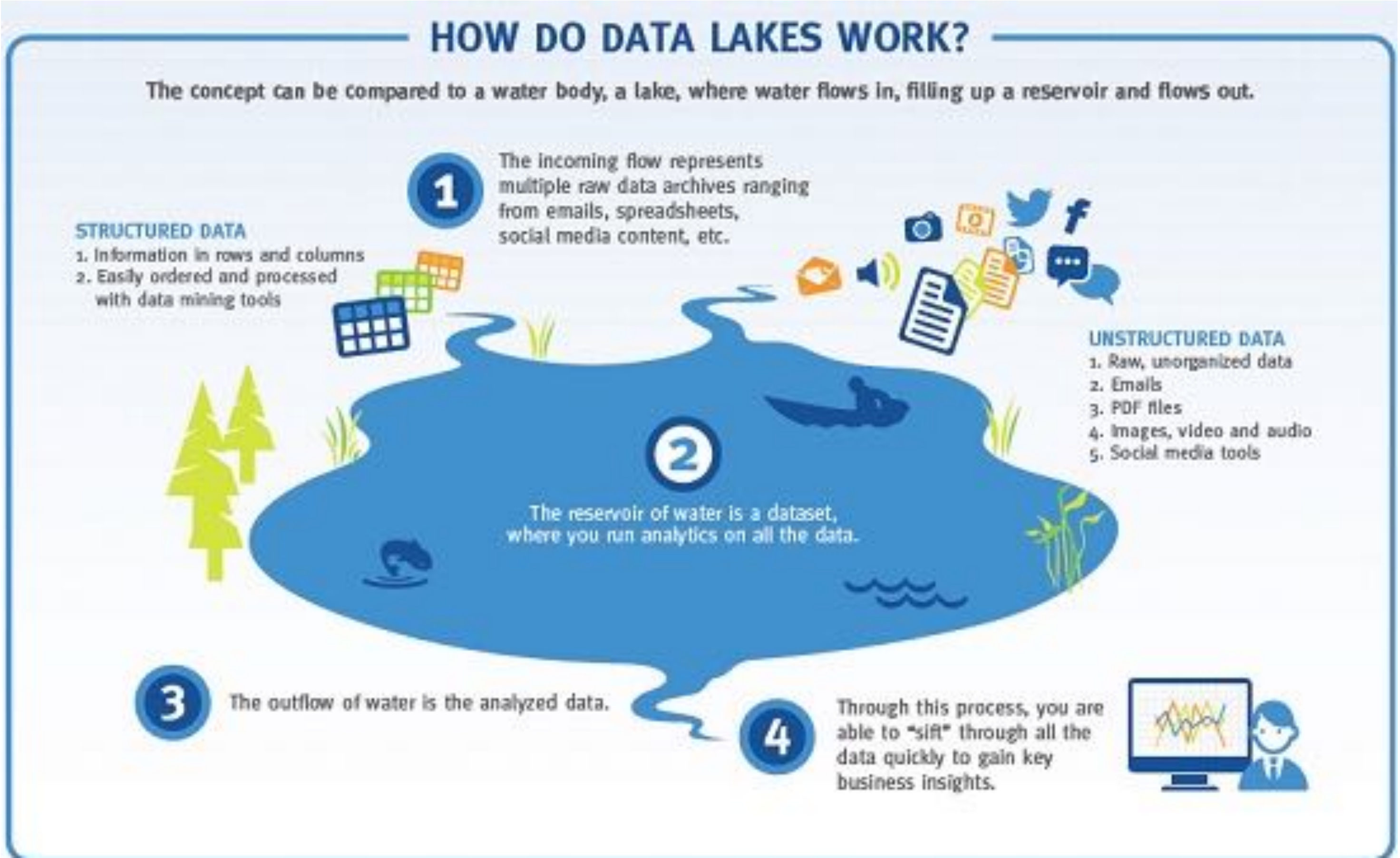
## Sources



## Structured, relational data



# NEW ARCHITECTURES – THE DATA LAKE



## HOW DO DATA LAKES WORK?

The concept can be compared to a water body, a lake, where water flows in, filling up a reservoir and flows out.

1 The incoming flow represents multiple raw data archives ranging from emails, spreadsheets, social media content, etc.

STRUCTURED DATA

1. Information in rows and columns
2. Easily ordered and processed with data mining tools

UNSTRUCTURED DATA

1. Raw, unorganized data
2. Emails
3. PDF files
4. Images, video and audio
5. Social media tools

2 The reservoir of water is a dataset, where you run analytics on all the data.

3 The outflow of water is the analyzed data.

4 Through this process, you are able to "sift" through all the data quickly to gain key business insights.

DATA SCIENCE PART TIME COURSE

# NO-SQL

# SQL/NoSQL

## SQL

- Traditional rows and columns data
- Strict structure / Primary Keys
- Entire column for each feature
- Industry standard

## NoSQL

- No well defined data structure
- Works better for unstructured data
- Cheaper hardware
- Popular among Startups

# SQL/NoSQL - Examples

## SQL

- MySQL
- Oracle
- Postgres
- SQLite
- SQLServer
- Redshift

## NoSQL

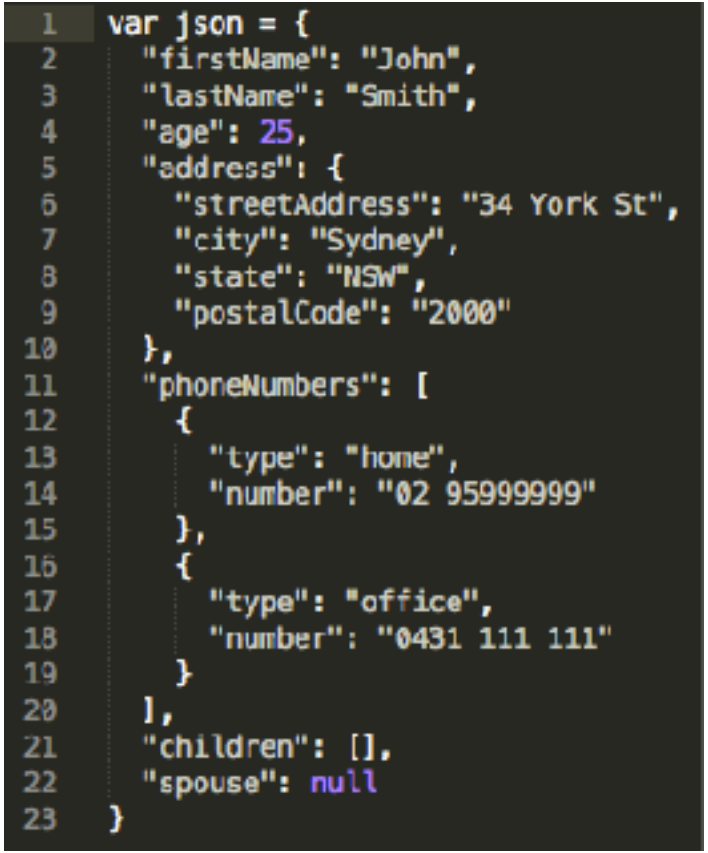
- MongoDB
- CouchDB
- Redis
- Cassandra
- Neo4j
- HBase

{JSON}

# JSON – WHAT IS IT?

## JSON - JavaScript Object Notation

- Human readable data with attribute-value pairs.
- What is inside the curly brackets is an object
- In the object we declare variables with ‘attribute’ : ‘value’ pairs



```
var json = {
  "firstName": "John",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "34 York St",
    "city": "Sydney",
    "state": "NSW",
    "postalCode": "2000"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "02 95999999"
    },
    {
      "type": "office",
      "number": "0431 111 111"
    }
  ],
  "children": [],
  "spouse": null
}
```

# JSON – HOW DOES IT RELATE TO DATA SCIENCE?

- Webservices provide application programming interfaces (APIs) are now usually transferring data via JSON
- Underlying document databases like MongoDB
- Increasingly common data format

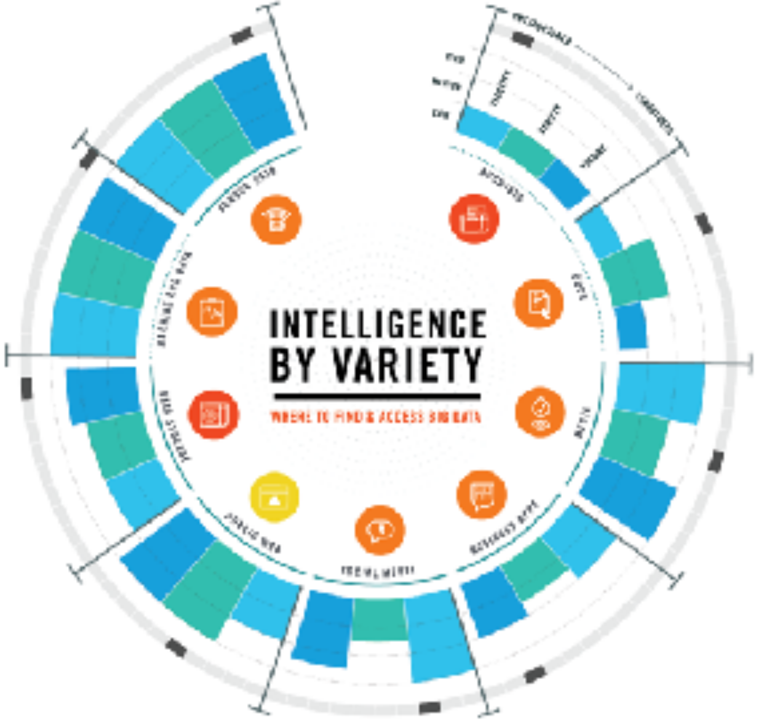
DATA SCIENCE PART TIME COURSE

# BIG DATA

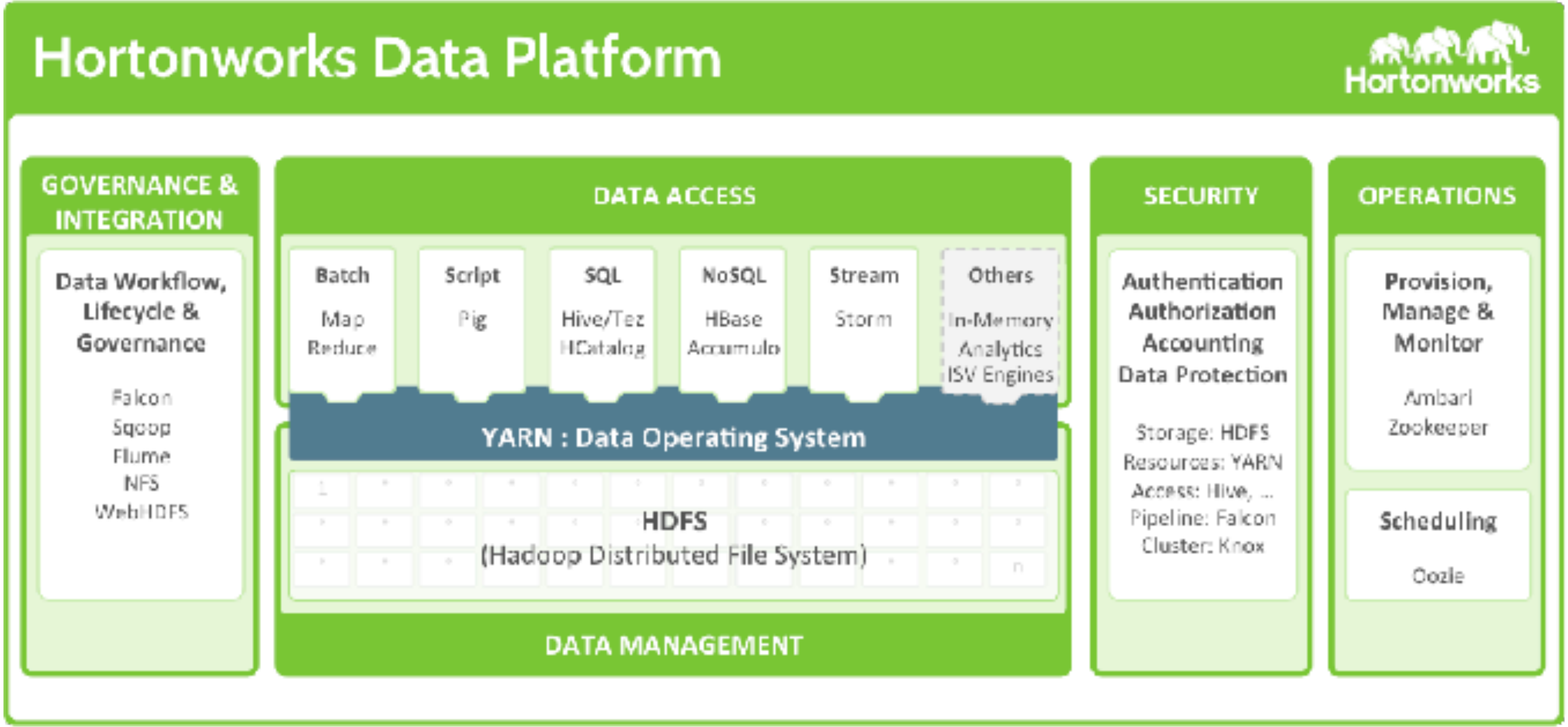
# BIG DATA SOURCES

Data comes from people, technology systems and sensors in the environment

- Organisations web applications
- External web applications (APIs)
- Devices with sensors (Infrastructure, Internet of Things)



# Hortonworks Stack



Hortonworks Data Platform

Hortonworks

GOVERNANCE & INTEGRATION

Data Workflow, Lifecycle & Governance

Falcon
Sqoop
Flume
NFS
WebHDFS

DATA ACCESS

Batch: Map Reduce

Script: Pig

SQL: Hive/Tez, HCatalog

NoSQL: HBase, Accumulo

Stream: Storm

Others: In-Memory, Analytics, ISV Engines

YARN : Data Operating System

HDFS
(Hadoop Distributed File System)

DATA MANAGEMENT

SECURITY

Authentication
Authorization
Accounting
Data Protection

Storage: HDFS
Resources: YARN
Access: Hive, ...
Pipeline: Falcon
Cluster: Knox

OPERATIONS

Provision, Manage & Monitor

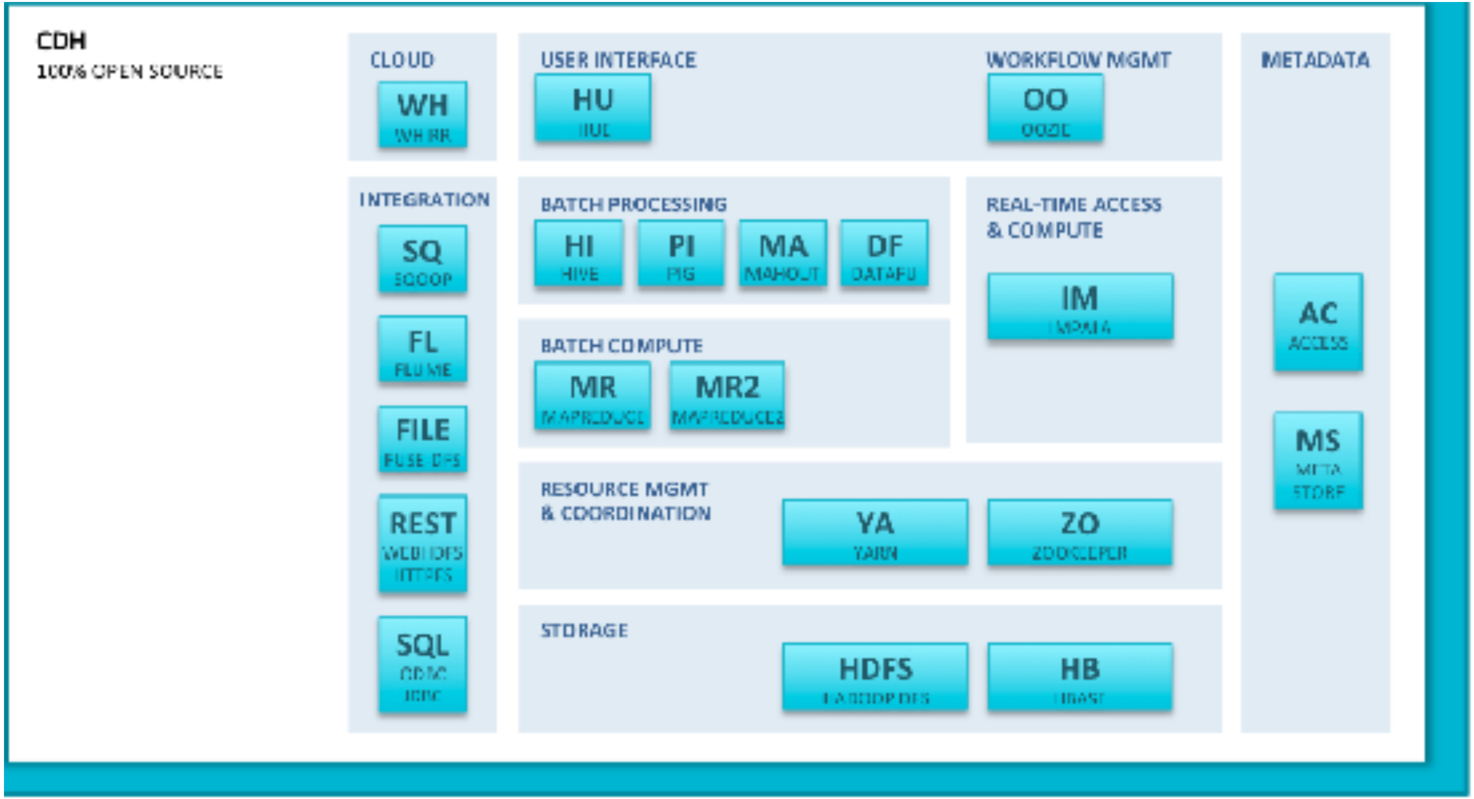
Ambari
Zookeeper

Scheduling

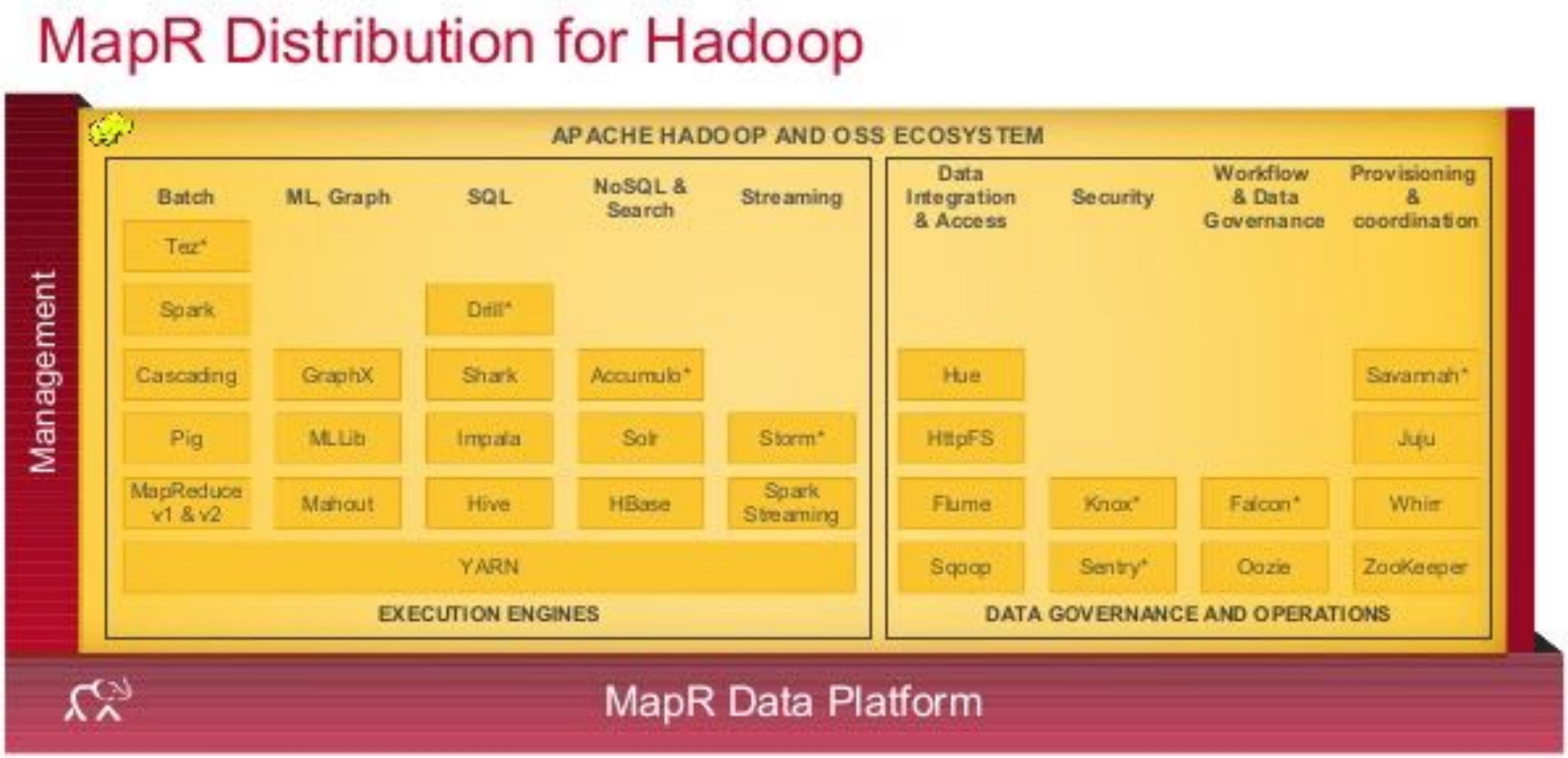
Oozie

# Cloudera Stack

cloudera



# MapR Stack



## MapR Distribution for Hadoop

APACHE HADOOP AND OSS ECOSYSTEM

| Batch | ML, Graph | SQL | NoSQL & Search | Streaming |
|---|---|---|---|---|
| Tez* | | | | |
| Spark | | Drill* | | |
| Cascading | GraphX | Shark | Accumulo* | |
| Pig | MLLib | Impala | Solr | Storm* |
| MapReduce v1 & v2 | Mahout | Hive | HBase | Spark Streaming |

YARN

EXECUTION ENGINES

| Data Integration & Access | Security | Workflow & Data Governance | Provisioning & coordination |
|---|---|---|---|
| Hue | | | Savannah* |
| HttpFS | | | Juju |
| Flume | Knox* | Falcon* | Whirr |
| Sqoop | Sentry* | Oozie | ZooKeeper |

DATA GOVERNANCE AND OPERATIONS

MapR Data Platform

# BIG DATA FILE FORMATS

Avro stores the data definition in JSON format making it easy to read and interpret, the data itself is stored in binary format making it compact and efficient. Avro files include markers that can be used to splitting large data sets into subsets suitable for MapReduce processing.

a new columnar storage format for Hadoop. Parquet can handle complex nested data structures.

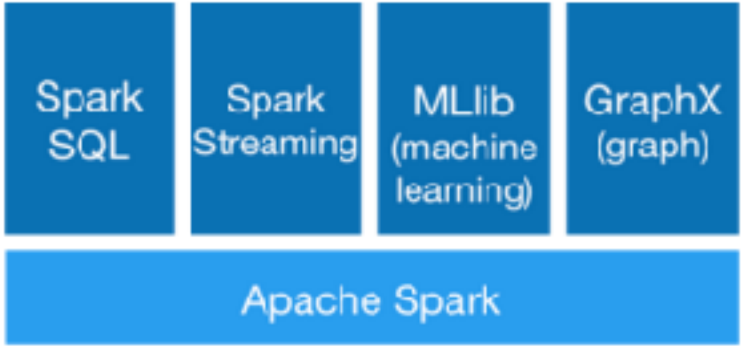
Optimized Row Columnar (ORC). A self-describing type-aware columnar file format designed for Hadoop ecosystem workloads. The columnar format lets the reader read, decompress, and process only the columns that are required for the current query.

DATA SCIENCE PART TIME COURSE

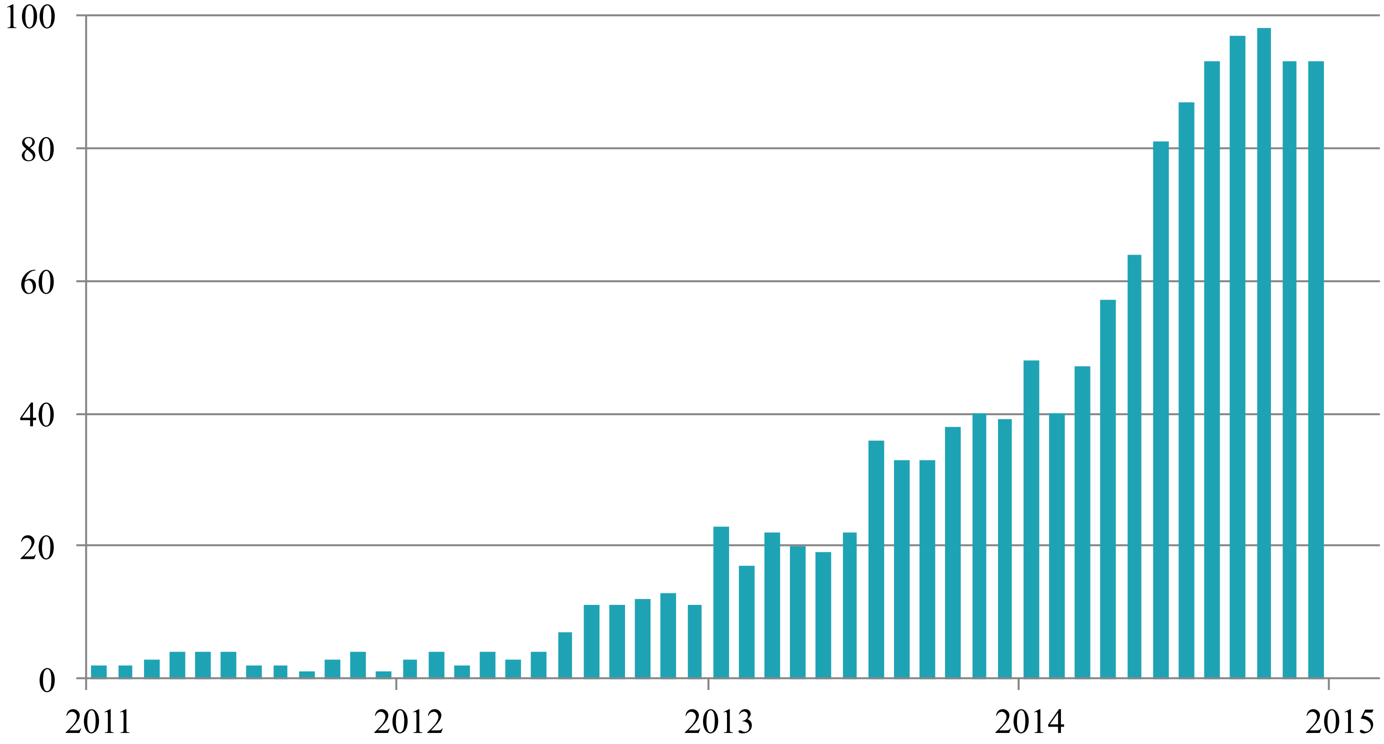
# SPARK

# SPARK – WHAT IS IT?

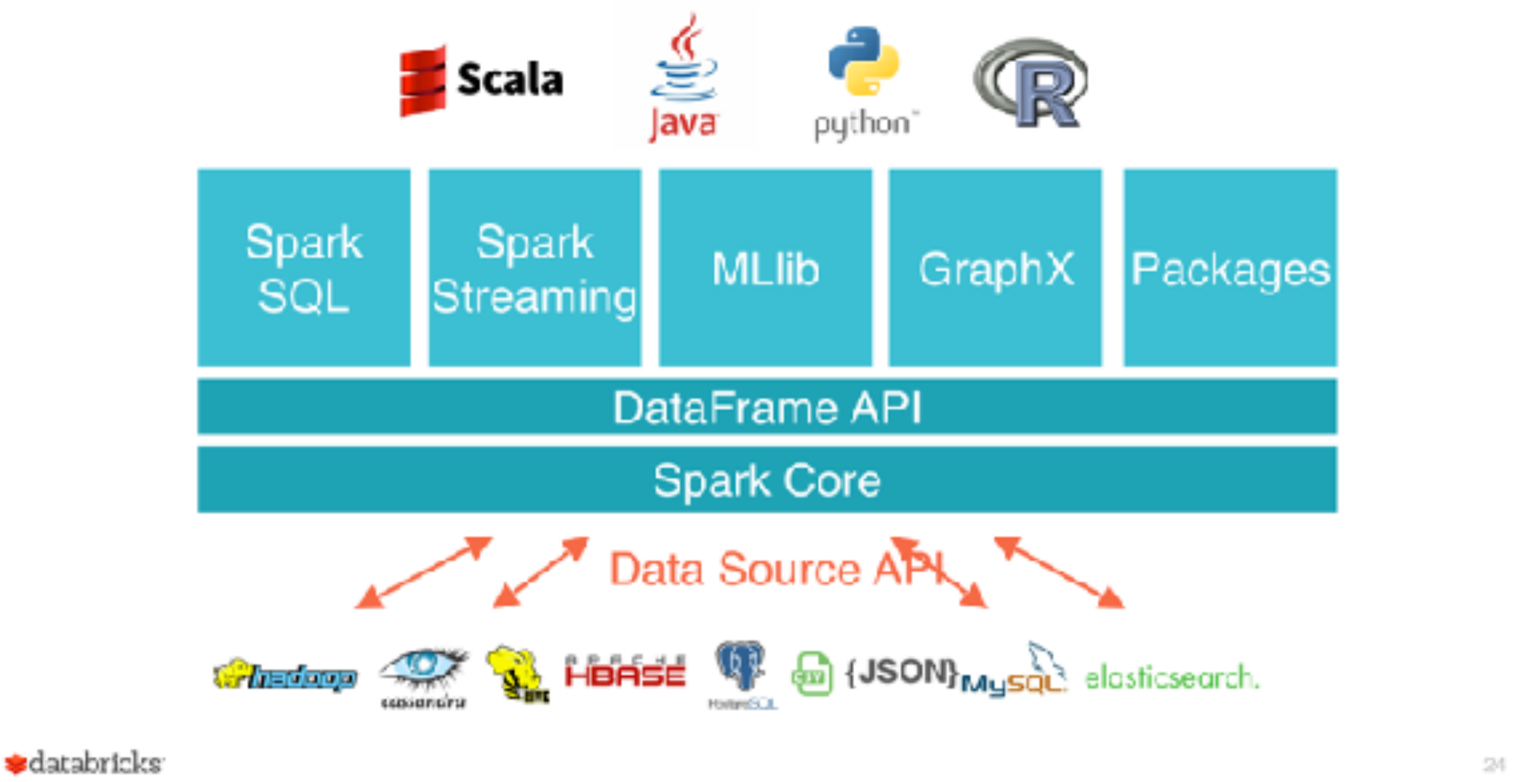
Spark is a fast and general processing engine compatible with Hadoop data. It can process data in HDFS, HBase, Cassandra, Hive, and any Hadoop InputFormat. It is designed to perform both batch processing (similar to MapReduce) and new workloads like streaming, interactive queries, and machine learning.



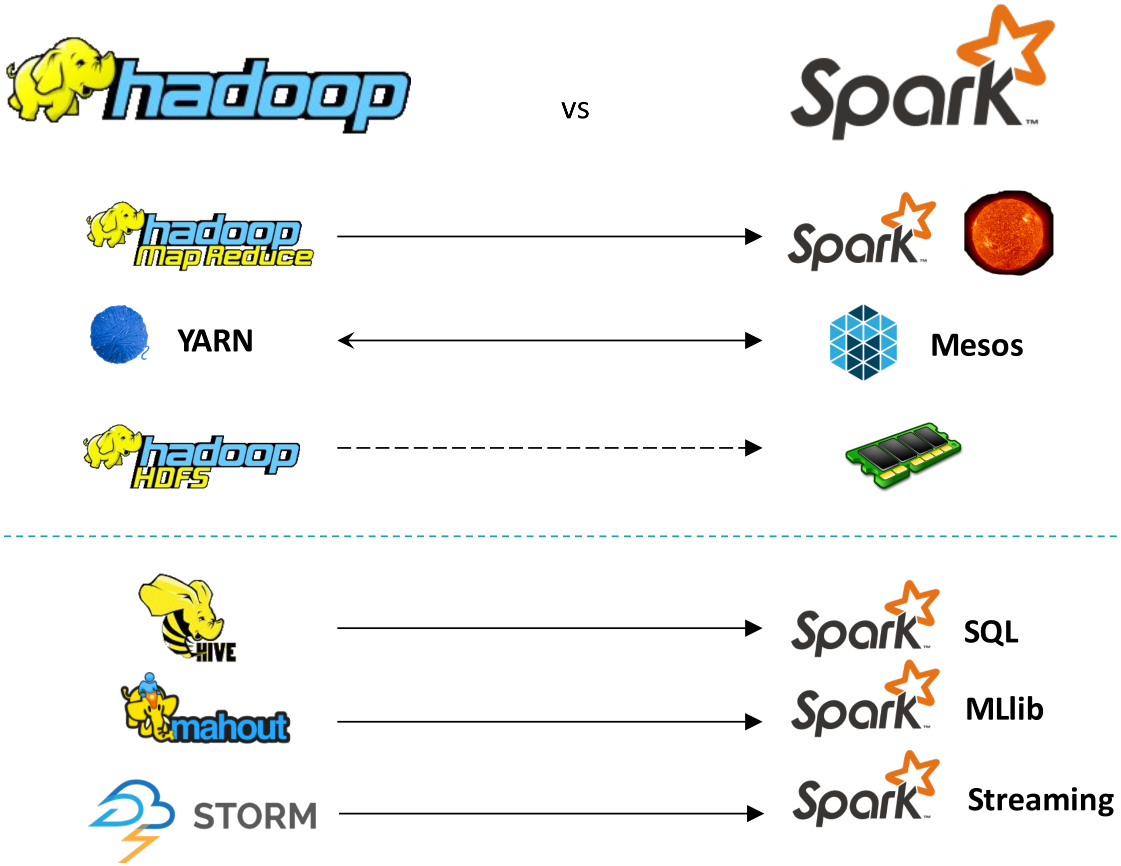
# SPARK - CONTRIBUTORS PER MONTH



# SPARK – WHAT IS IT?



# SPARK – WHAT IS IT?



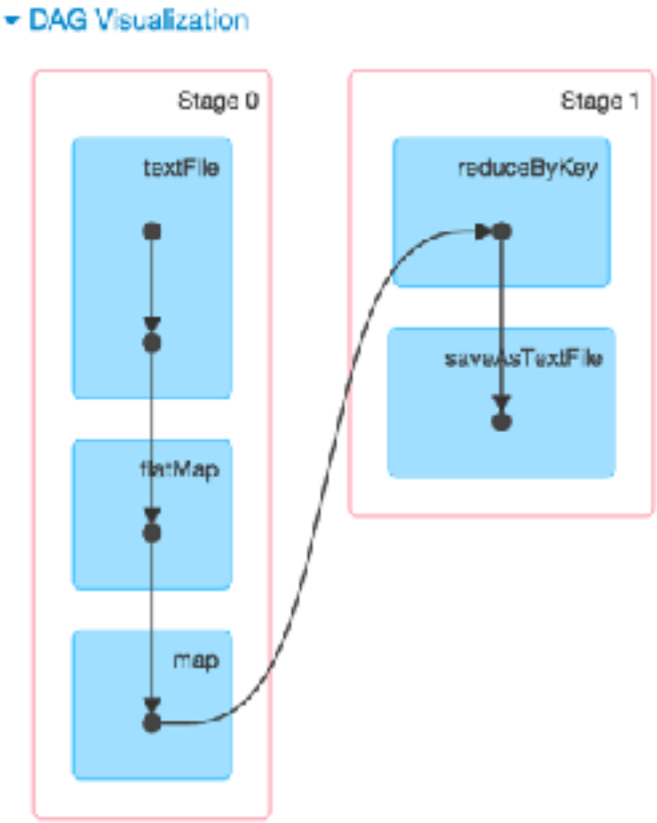
## RESILIENT DISTRIBUTED DATASETS (RDDs)

Spark revolves around the concept of a resilient distributed dataset (RDD), which is a fault-tolerant collection of elements that can be operated on in parallel.

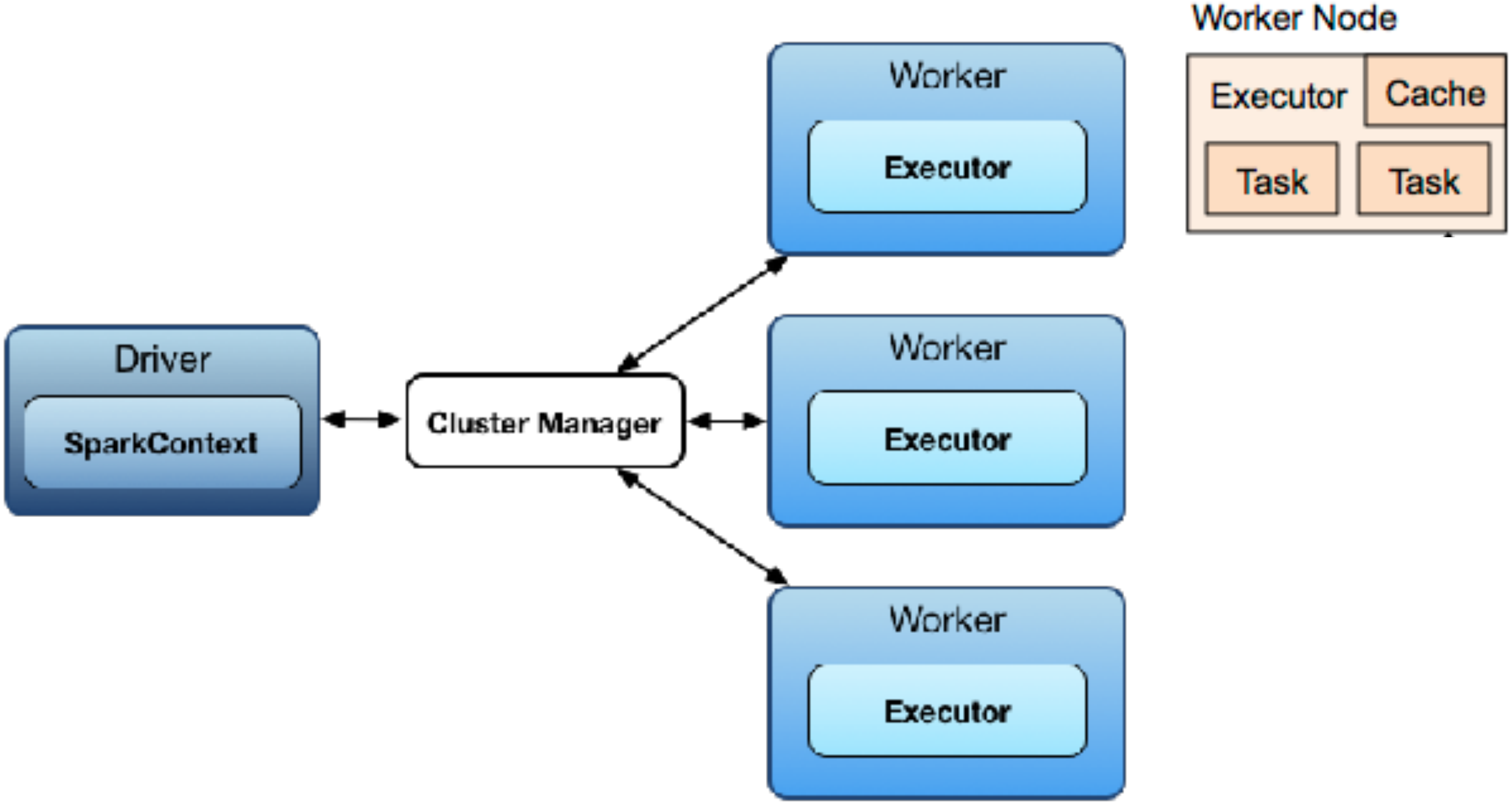
There are two ways to create RDDs:

1. Parallelizing an existing collection in your driver program
2. Referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat

# Lineage

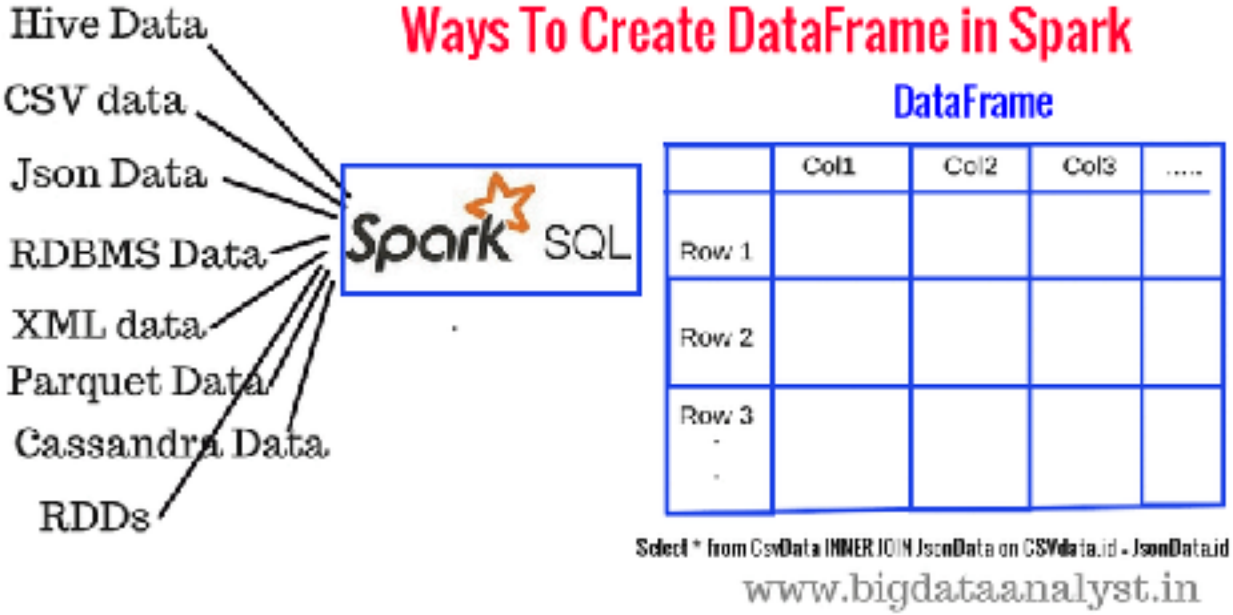

DAG Visualization
Stage 0
textFile
flatMap
map
Stage 1
reduceByKey
saveAsTextFile

# SPARK CLUSTER



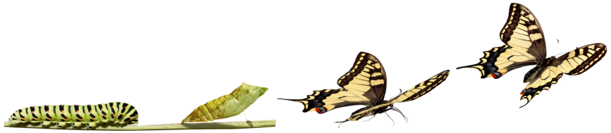
# Dataframes

DataFrames API was inspired by data frames in R and Pandas in Python. DataFrames integrates with Python, Java, Scala and R.



# Spark Operations

Spark Operations =

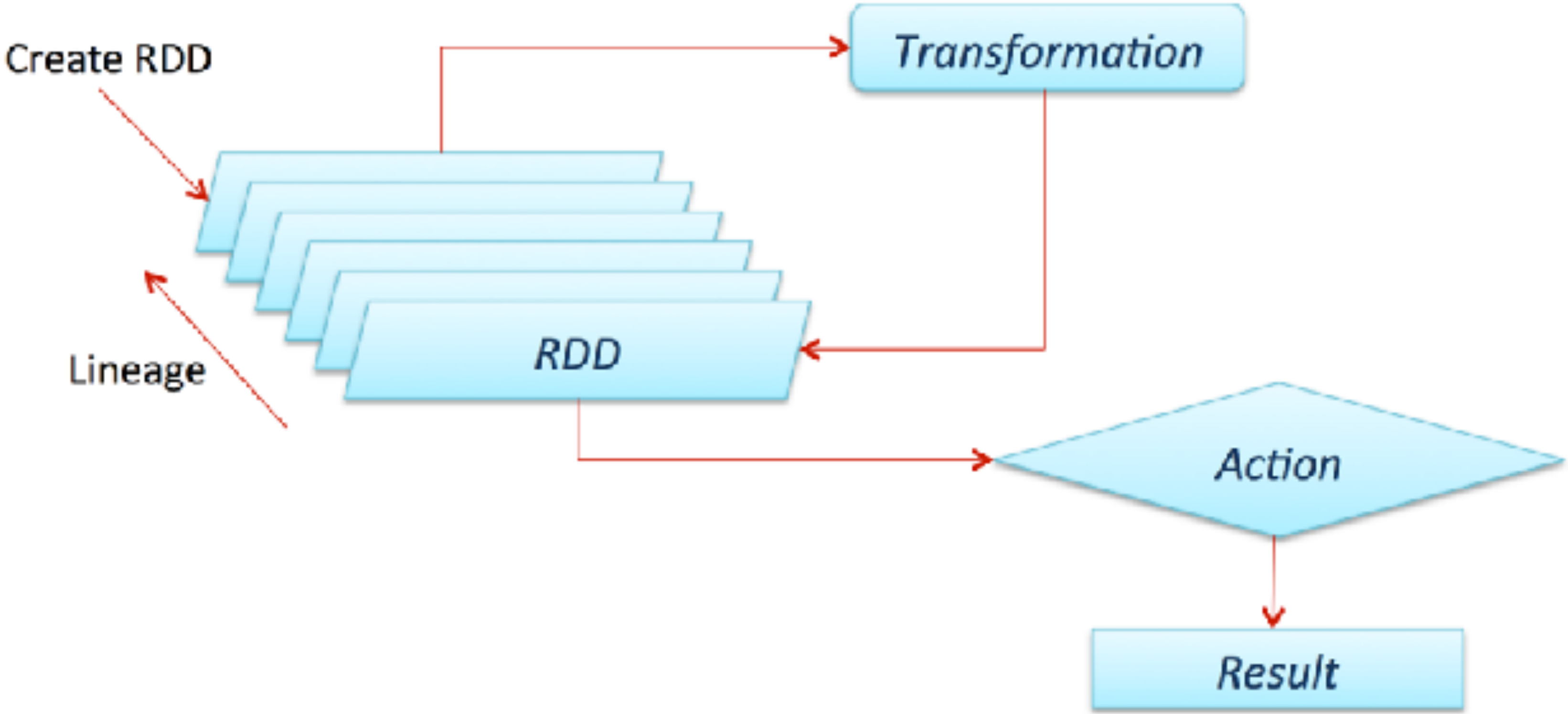


TRANSFORMATIONS

+

ACTIONS

## Spark Operations



# Spark Operations

⬜ = easy 🟦 = medium

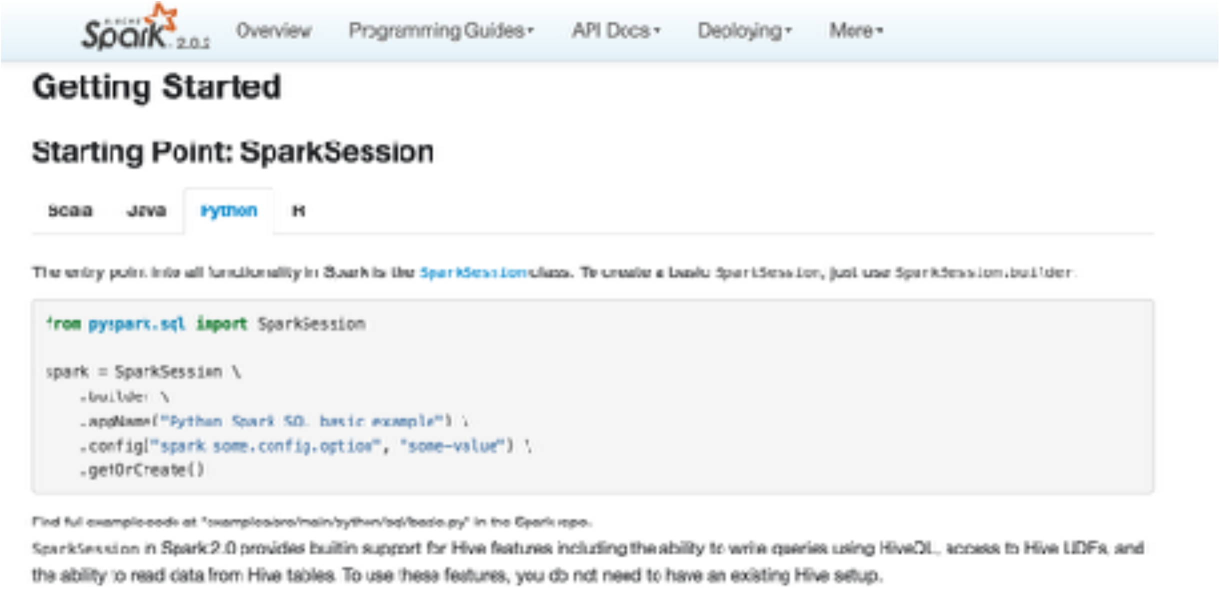
## Essential Core & Intermediate Spark Operations

| | General | Math / Statistical | Set Theory / Relational | Data Structure / I/O |
|---|---|---|---|---|
| TRANSFORMATIONS | map<br>filter<br>flatMap<br>mapPartitions<br>mapPartitionsWithIndex<br>groupBy<br>sortBy | sample<br>randomSplit | union<br>intersection<br>subtract<br>distinct<br>cartesian<br>zip | keyBy<br>zipWithIndex<br>zipWithUniqueID<br>zipPartitions<br>coalesce<br>repartition<br>repartitionAndSortWithinPartitions<br>pipe |
| ACTIONS | reduce<br>collect<br>aggregate<br>fold<br>first<br>take<br>forEach<br>top<br>treeAggregate<br>treeReduce<br>forEachPartition<br>collectAsMap | count<br>takeSample<br>max<br>min<br>sum<br>histogram<br>mean<br>variance<br>stdev<br>sampleVariance<br>countApprox<br>countApproxDistinct | takeOrdered | saveAsTextFile<br>saveAsSequenceFile<br>saveAsObjectFile<br>saveAsHadoopDataset<br>saveAsHadoopFile<br>saveAsNewAPIHadoopDataset<br>saveAsNewAPIHadoopFile |

# Spark Documentation

Use the programming guides on the Spark Website:

http://spark.apache.org/docs/latest/programming-guide.html



## Getting Started

### Starting Point: SparkSession

The entry point into all functionality in Spark is the SparkSession class. To create a basic SparkSession, just use SparkSession.builder:

```
from pyspark.sql import SparkSession

spark = SparkSession \
    .builder \
    .appName("Python Spark SQL basic example") \
    .config("spark.some.config.option", "some-value") \
    .getOrCreate()
```

Find full example code at "examples/src/main/python/sql/basic.py" in the Spark repo.

SparkSession in Spark 2.0 provides builtin support for Hive features including the ability to write queries using HiveQL, access to Hive UDFs, and the ability to read data from Hive tables. To use these features, you do not need to have an existing Hive setup.

You can select which programming language API you want to write Spark code with :

- Scala
- Jave
- Python (PySpark)
- R (SparkR)

# SPARK MLLIB

**Data types**

**Basic statistics**

- summary statistics
- correlations
- stratified sampling
- hypothesis testing
- streaming significance testing
- random data generation

**Classification and regression**

- linear models (SVMs, logistic regression, linear regression)
- naive Bayes
- decision trees
- ensembles of trees (Random Forests and Gradient-Boosted Trees)
- isotonic regression

**Collaborative filtering**

- alternating least squares (ALS)

**Clustering**

- k-means
- Gaussian mixture
- power iteration clustering (PIC)
- latent Dirichlet allocation (LDA)
- bisecting k-means
- streaming k-means

**Dimensionality reduction**

- singular value decomposition (SVD)
- principal component analysis (PCA)

**Feature extraction and transformation**

**Frequent pattern mining**

- FP-growth
- association rules
- PrefixSpan

**Evaluation metrics**

**PMML model export**

**Optimization (developer)**

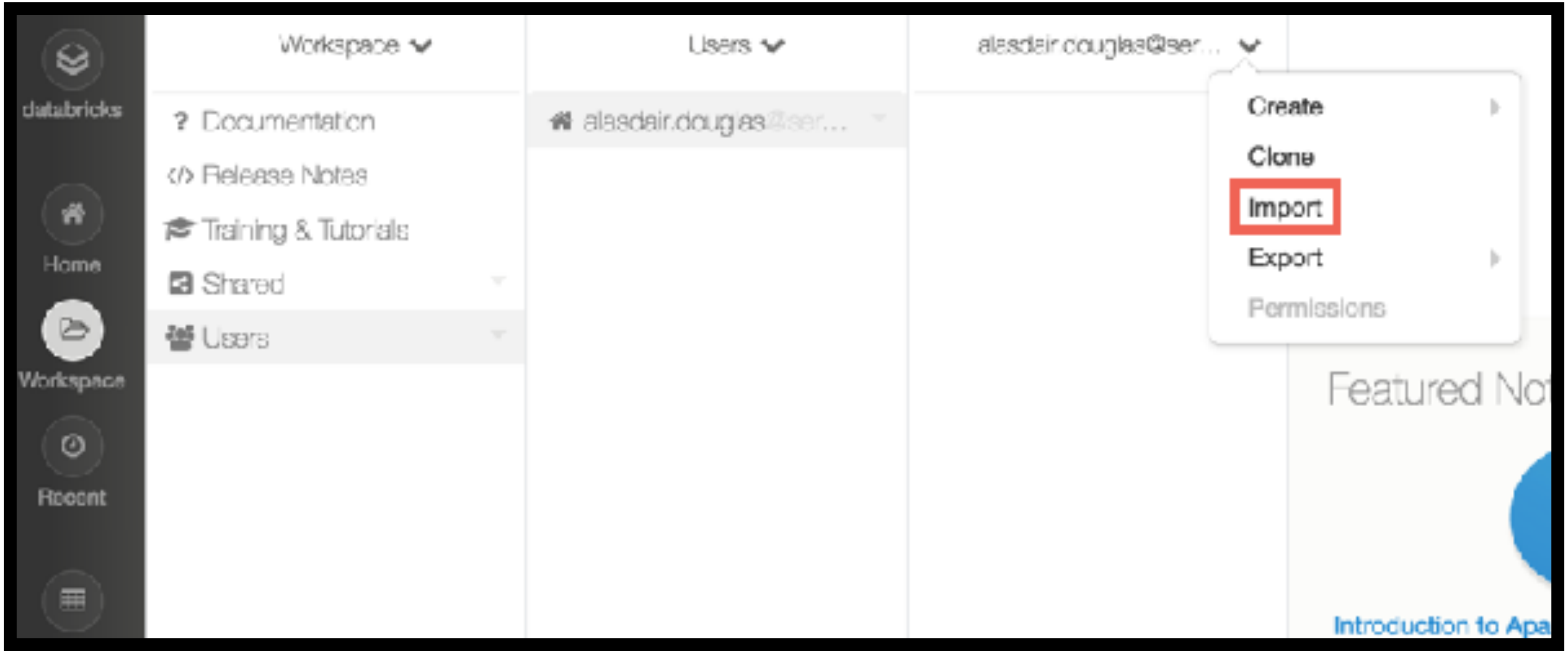
# COURSE FEEDBACK

## DATA SCIENCE – Week 8 Day 1

# LAB

- Log into Databricks community edition https://databricks.com/try-databricks
- https://docs.databricks.com/_static/notebooks/gentle-introduction-to-apache-spark.html

# In databricks import the notebook from the below URL



https://docs.databricks.com/_static/notebooks/gentle-introduction-to-apache-spark.html

# SYNCHING YOUR FORK WITH THE COURSE REPO

1. re-name your labs with lab_name.<yourname>.ipynb  (to prevent a conflict)
2. cd <path to the root of your SYD_DAT_6 local repo>
3. commit your changes ahead of sync
   - git status
   - git add .
   - git commit -m “descriptive label for the commit”
   - git status
4. download new material from official course repo (upstream) and merge it
   - git checkout master  (ensures you are in the master branch)
   - git fetch upstream
   - git merge upstream/master

# HOMEWORK

**Homework**

- Read Natural Language Processing website – http://www.nltk.org/ (5 mins)
- Read and be able to explain one use case of the Alchemy API (10 mins)
- Download and install NLTK for Python (10mins)

**Reading**

- For those that want a good foundation in data stores and architecture, the redbook 5th edition is a good reference http://www.redbook.io/