



**BIG DATA**

## **Hadoop Ecosystem**

---

**K V Subramaniam**

Computer Science and Engineering

# Hadoop Ecosystem Overview



What we have learnt so far..

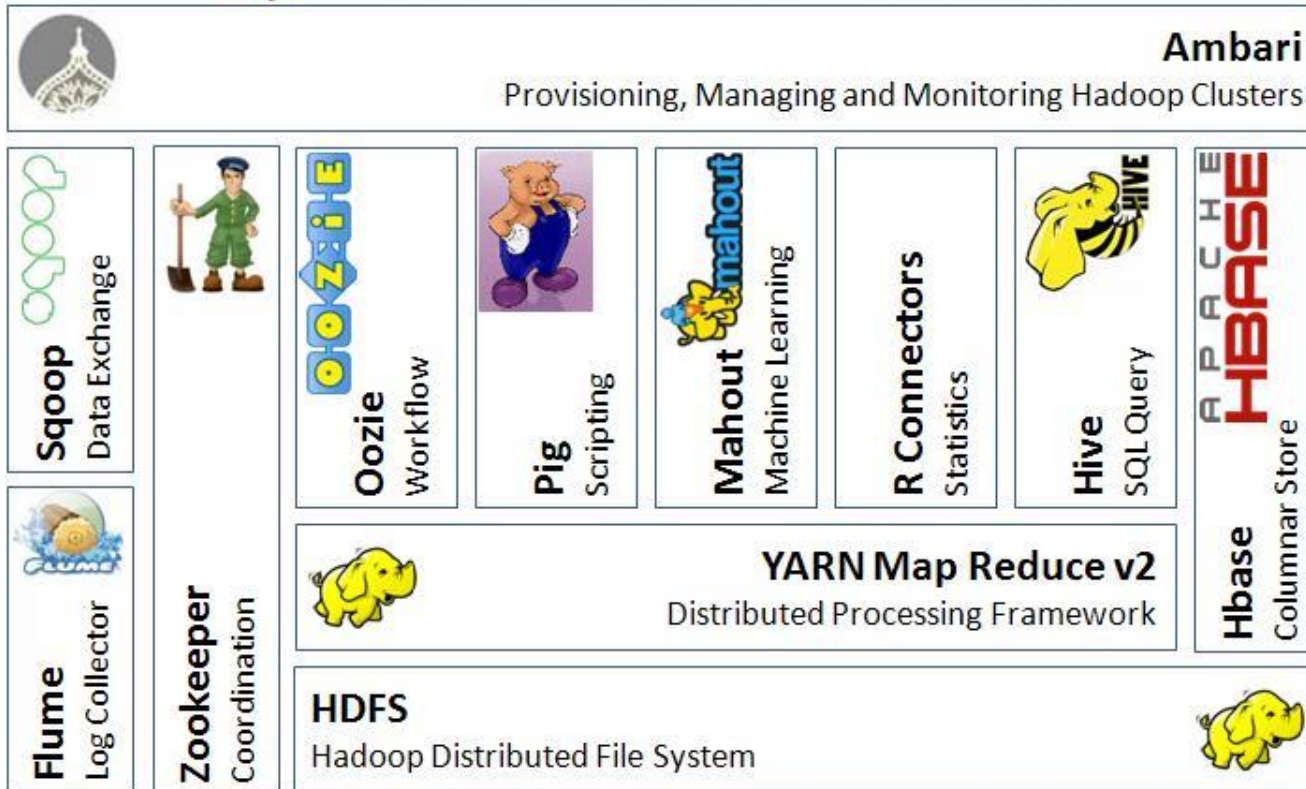
HDFS – for storage

And MapReduce (Hadoop) for computation

So where does it all fit in the bigger scheme of a  
Big Data architecture



## Apache Hadoop Ecosystem



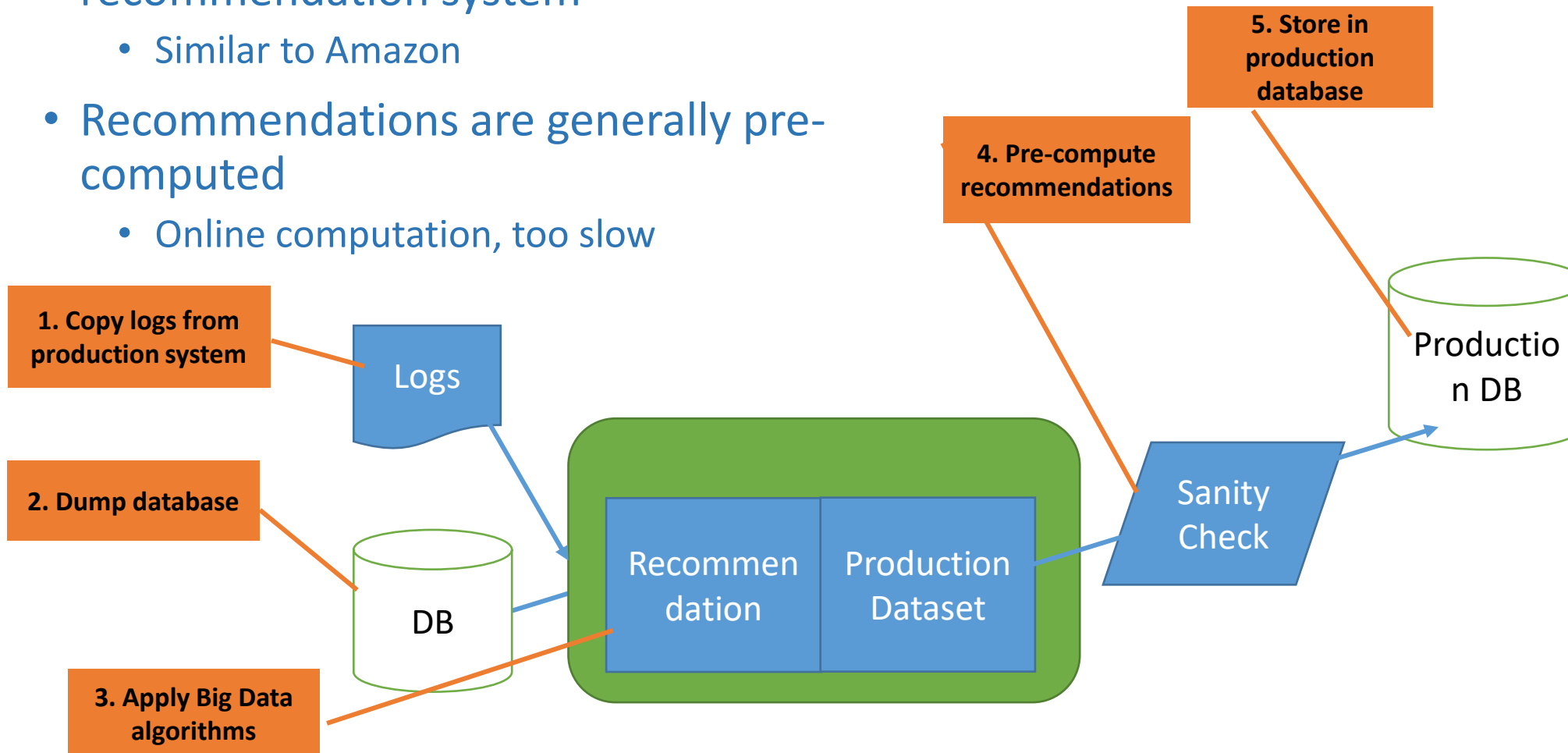
## Hadoop Workflows: Oozie

- Suppose we want to build a recommendation system like in AMAZON
  - What inputs would we need?
  - How often would we need to update the recommendations (every hour? Every day? EVERY week?)
  - Is it enough to just build a recommendation algorithm?
    - Let's assume that we have a MAGIC **"RECOMMENDATION ALGORITHM"** that can work PROVIDED the right inputs are given to it.

# BIG DATA

## Hadoop Ecosystem

- We want to use the sales to build a recommendation system
  - Similar to Amazon
- Recommendations are generally pre-computed
  - Online computation, too slow



# BIG DATA

## Workflow Definition

---

- The sequence of steps is called a *Workflow*
- Workflows are common in data centers
- Users would like to
  - Specify the steps
  - Specify when the steps are to be run
    - Maybe periodically
  - Run the Workflow
  - What to do in case of error

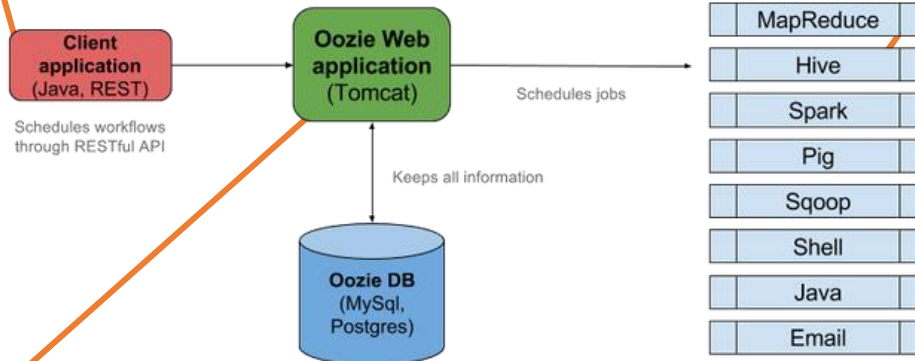


# BIG DATA

## Oozie Architecture

1. User submits  
Workflow (XML)

2. Oozie: Tomcat  
application



3. Some of the  
applications supported  
by Oozie

4. Workflow can  
include any of these  
applications

Image courtesy:

<https://oymolenko.blog/2017/10/01/scheduling-jobs-in-hadoop-through-oozie/>

# BIG DATA

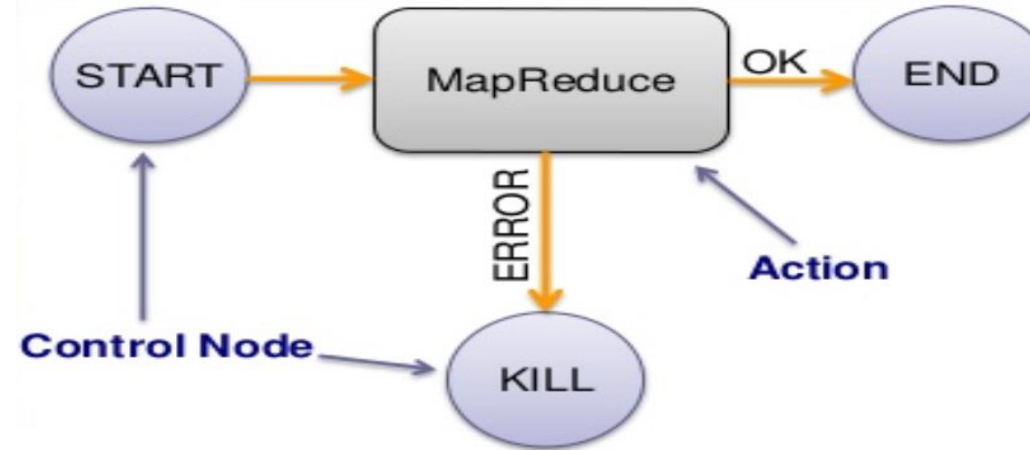
## Oozie Workflow (pictorial)

- **Action nodes**

- Does something, e.g., run Mapreduce
- Every action node has a normal exit and an error exit

- **Control nodes**

- Start is the beginning of a Workflow
- End and kill are the end
- Other control nodes
  - Fork (start parallel tasks) and Join (merge parallel tasks)
  - Decision: like switch



DAG Expressing Workflow

# BIG DATA

## Oozie References

---

- [T1 – Chapter 2.6.1.2](#)
- [T2 – Chapter 7.5](#)
- <http://oozie.apache.org/docs/4.3.0/index.html>
  - Official Oozie Homepage
- <https://overmolenko.blog/2017/10/01/scheduling-jobs-in-hadoop-through-oozie/>
  - A very good introduction to Oozie

## Hadoop Workflows: Ambari

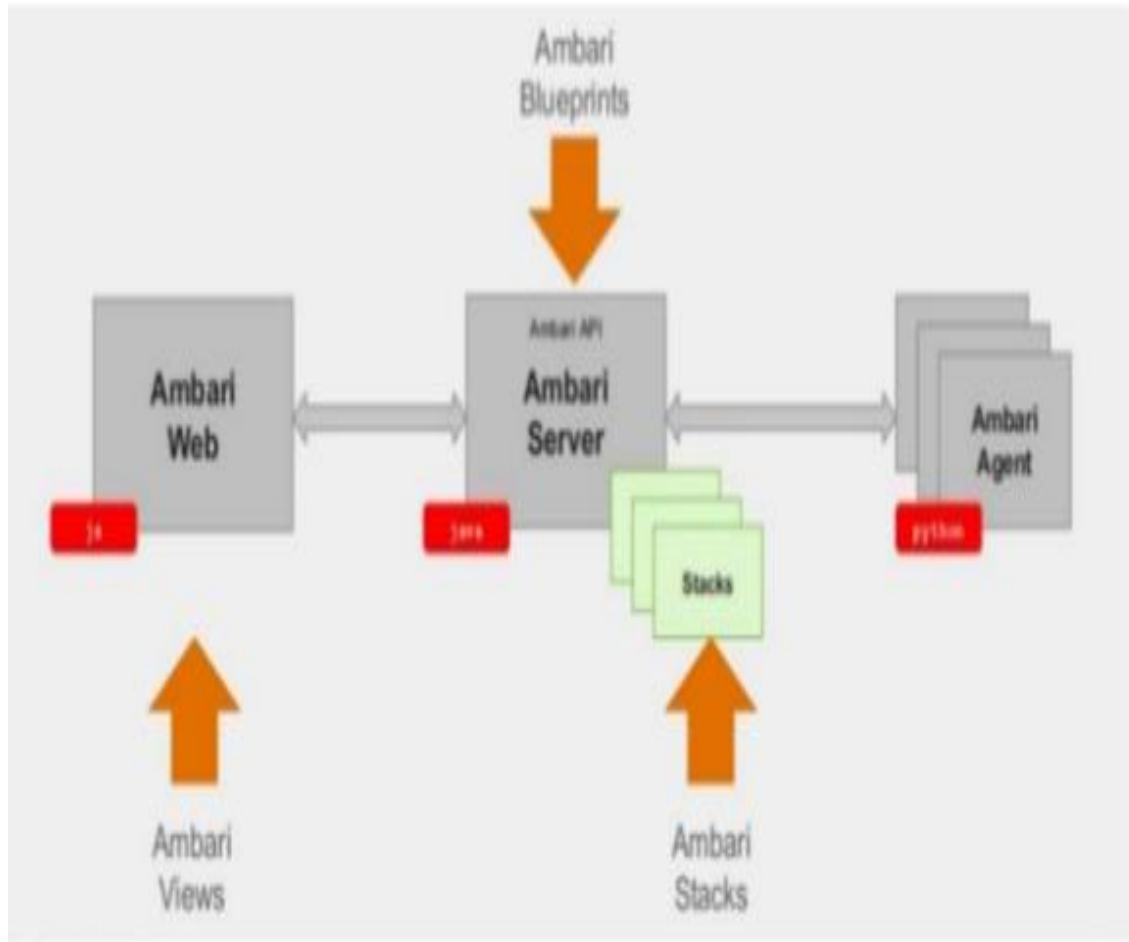


# BIG DATA

## Ambari : Deploy and Manage Hadoop Clusters

---

- Simplifies Installation, Configuration and Management
- Easy, efficient, repeatable creation of clusters
- Manages and Monitors clustering



- **Ambari Stacks**  
Describes the applications to be installed, eg, Hadoop, its components and its structure
- **Ambari Blueprints**  
Creation of the cluster
- **Ambari Views**  
User interface

# BIG DATA

## Ambari Stacks

---

What do we need to define in Ambari to install a cluster ?

Term	Meaning
Stack	Set of services, where to get the software packages, e.g. HDP (Hortonworks Data Platform)
Service	Components that make up the service e.g, HDFS
Component	Building blocks of the service – Namenode, Datanode
Category	Master, slave, client

- 1 Create Cluster
- 2 Add Nodes
- 3 Select Services
- 4 Assign Hosts
- 5 Select Mount Points
- 6 Custom Config
- 7 Review & Deploy

### Which nodes do you want to install Hadoop on?

We will use the SSH private key for the `root` user and a file containing a list of hostnames to perform installation on your nodes. The corresponding public key must already be in `authorized_keys` on all the nodes.

SSH Private Key File for `root`

Hosts File (newline-delimited list of hostnames)

☐ Use local yum mirror instead of downloading packages from the internet



1 Create Cluster Deploy

### Node Discovery and Preparation

Finding reachable nodes: All 5 nodes succeeded  
Obtaining information about reachable nodes: All 5 nodes succeeded  
Verifying and updating node information: All 5 nodes succeeded  
Preparing discovered nodes: All 5 nodes succeeded  
Finalizing bootstrapped nodes: All 5 nodes succeeded

Finished node discovery and preparation. Proceed to Select Services

Which node...  
We will use the S...  
already be in aut...  
Key File for ro...  
gate/Downloa...  
SSH Private...  
/Users/vgc...  
gate/Downloa...  
Hosts File (n...  
/Users/vgc...  
um mirror inst...  
☐ Use local S...  
Add Node

1 Create Cluster

2 Add Nodes

3 Select Services

4 Assign Hosts

5 Select Mount Points

6 Custom Config

7 Review & Deploy

### Which services do you want to install?

We will automatically take care of dependencies (e.g., HBase requires ZooKeeper, etc.)

☒ Select all

☒ HDFS - Apache Hadoop Distributed File System

☒ MapReduce - Apache Hadoop Distributed Processing Framework

☒ Ganglia - Ganglia-based Metrics Collection for HDP

☒ Nagios - Nagios-based Monitoring for HDP

☒ HBase - Apache HDFS-based Non-relational Distributed Database

☒ Pig - Platform for Analyzing Large Data Sets

☒ Sqoop - Tool for transferring bulk data between Apache Hadoop and structured datastores such as relational databases

☒ Oozie - Workflow/Coordination system to manage Apache Hadoop jobs

☒ Hive/HCatalog - Hive - Data Warehouse system for Apache Hadoop, HCatalog - Table and Storage Management service for data created using Apache Hadoop

☒ Templeton - Webservice APIs for Apache Hadoop

☒ ZooKeeper - Centralized Service for Configuration Management and Distribution Synchronization

Select Services

# BIG DATA

## Hadoop Ecosystem

- 1 Create Cluster
- 2 Add Nodes
- 3 Select Services
- 4 Assign Hosts
- 5 Select Mount Points
- 6 Custom Config
- 7 Finish

### Customize Settings

We have come up with reasonable default settings. Customize as you see fit.

#### HDFS

NameNode directories	<input type="text" value="/grid/0/hadoop/hdfs/namenode/"/>	
DataNode directories	<input type="text" value="/grid/0/hadoop/hdfs/data/grid/1/h"/>	
SecondaryNameNode Checkpoint directory	<input type="text" value="/grid/0/hadoop/hdfs/namesecond"/>	
WebHDFS enabled	<input type="checkbox"/>	
Hadoop maximum Java heap size	<input type="text" value="1536"/>	MB
NameNode Java heap size	<input type="text" value="2760"/>	MB
NameNode new generation size	<input type="text" value="200"/>	MB
Reserved space for HDFS	<input type="text" value="1"/>	GB
DataNode maximum Java heap size	<input type="text" value="1536"/>	MB
DataNode volumes failure toleration	<input type="text" value="0"/>	
HDFS Maximum Checkpoint Delay	<input type="text" value="21600"/>	seconds
HDFS Maximum Edit Log Size for Checkpointing	<input type="text" value="0.5"/>	GB

# BIG DATA

## Hadoop Ecosystem

### Deployment Progress

Cluster install	<div></div>	Completed
HDFS start	<div></div>	Completed
HDFS test	<div></div>	Completed
<b>MapReduce start</b>	<div></div>	<b>In Progress</b>
MapReduce test	<div></div>	Pending
ZooKeeper start	<div></div>	Pending
ZooKeeper test	<div></div>	Pending
HBase start	<div></div>	Pending
HBase test	<div></div>	Pending
Pig test	<div></div>	Pending
Sqoop test	<div></div>	Pending
Oozie start	<div></div>	Pending
Oozie test	<div></div>	Pending
Hive/HCatalog start	<div></div>	Pending
Hive/HCatalog test	<div></div>	Pending
Templeton start	<div></div>	Pending
Templeton test	<div></div>	Pending
Dashboard start	<div></div>	Pending
Ganglia start	<div></div>	Pending
Nagios start	<div></div>	Pending



**Pig : Building High-Level Dataflows  
over Map Reduce**

# BIG DATA

## Disadvantages of MapReduce

---

- For Data analysis, Map-Reduce is too low-level
- Writing Map and Reduce code requires retraining.
- Something SQL-Like may be better
  - HIVE is an option (which we will look at later)
  - But how about a scripting language

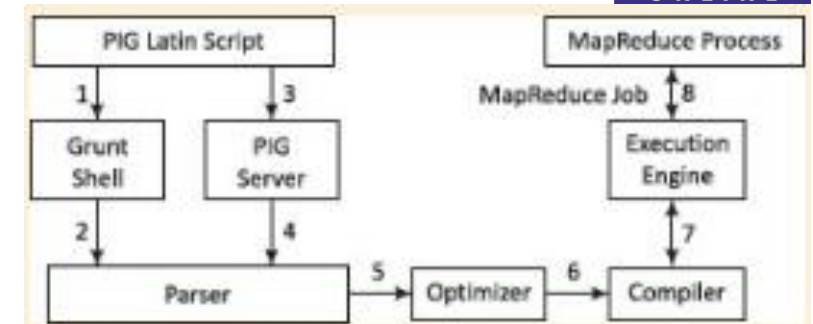
# BIG DATA

## PIG Introduction

- Complex data transformations using scripts
  - Builtin operators – join, group, filter, limit...
- Interactive shell → Grunt
- Language → Pig Latin
- Scripts are internally converted to Map Reduce jobs.
- Created @Yahoo



**PES**  
UNIVERSITY  
ONLINE



# BIG DATA

## Example Data Analysis Task



- Find the top 10 most popular IPL matches in each venue.

**Visits**

User	matchId	Time
RajniKan	Match 400	2:00
RajniKan	Match 201	5:00
Superman	Match 42	10:05
Spiderman	Match 108	13:03



**MatchInfo**

matchId	Venue	Winner
Match 108	Chennai	CSK
Match 201	Bengaluru	RCB
Match 42	Kolkata	DC
Match 400	Mumbai	RCB

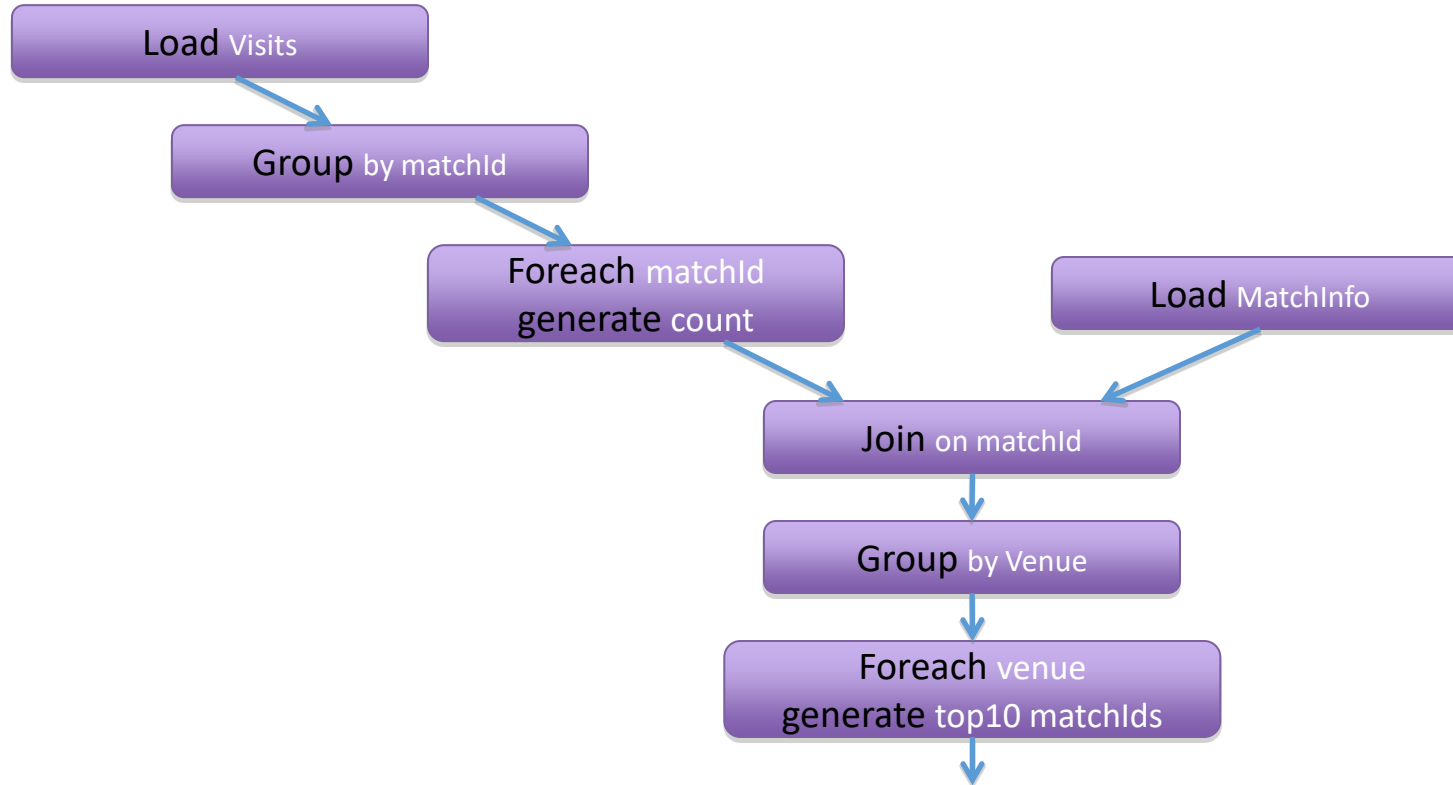




# BIG DATA

## Data Flow

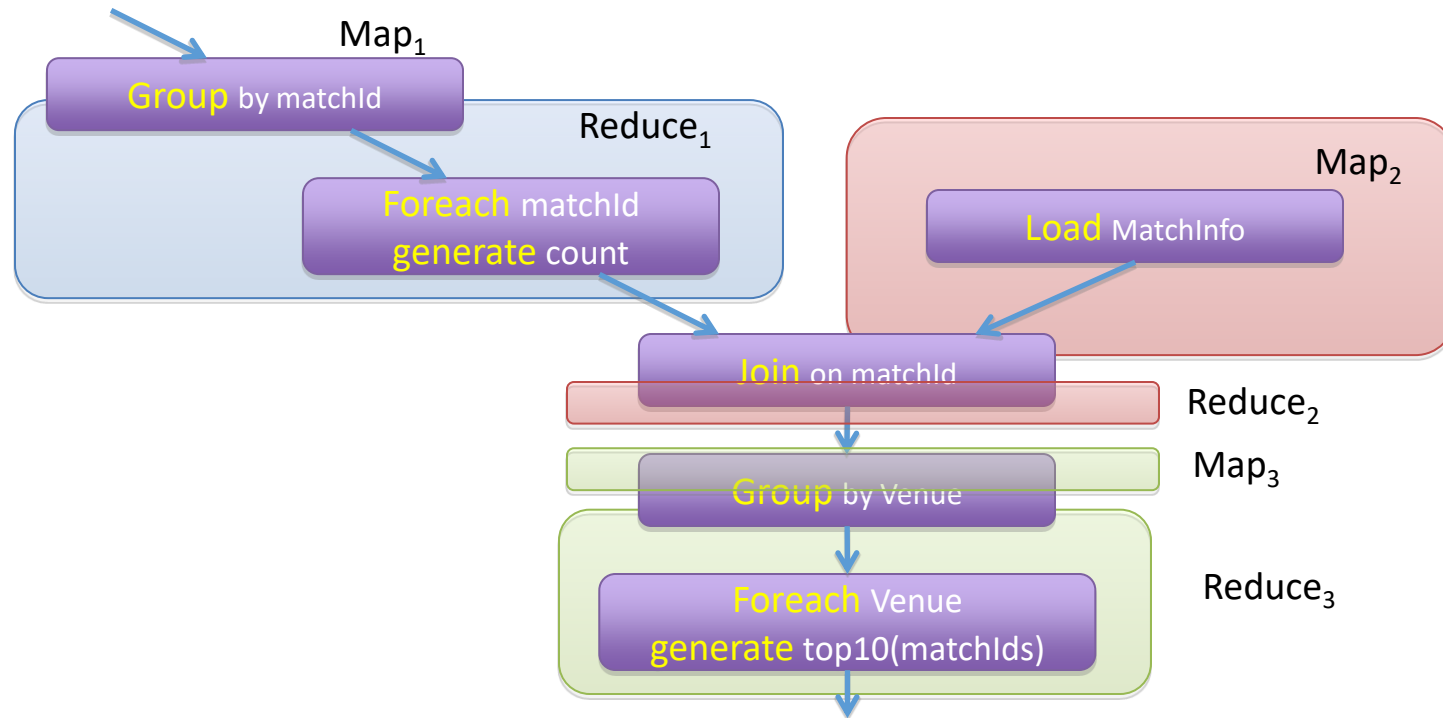
---



# BIG DATA

## Hadoop Ecosystem

### Compilation into Map-Reduce

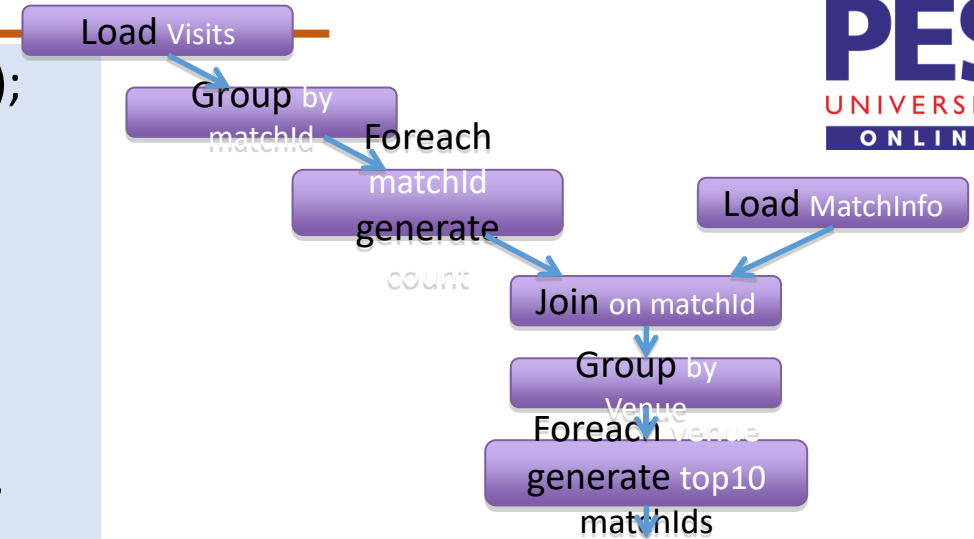


# BIG DATA

## In Pig Latin



- visits = load '/ipldata/visits' as (user,matchid, time);
- gMatches = group visits by matchId;
- matchPopularity = foreach gMatches generate matchId, count(visits);
- matchInfo = load '/ipldata/matchInfo' as (url, venue, winner);
- venueCounts = join gMatches by matchId, matchInfo by matchId;
- gVenues = group venueCounts by venue;
- topMatches = foreach gVenues generate top(matchPopularity,10);
- store topMatches into '/data/topMatches' ;



# BIG DATA

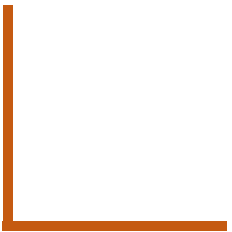
## Pig References

---

- [T1 – Chapter 4.6 – you are not expected to memorize the syntax of Pig. Just the basics and how it is converted to map reduce tasks. Please go through the entire section if you want to learn to code.](#)
- [T2 – Chapter 7.5](#)



**SQOOP**



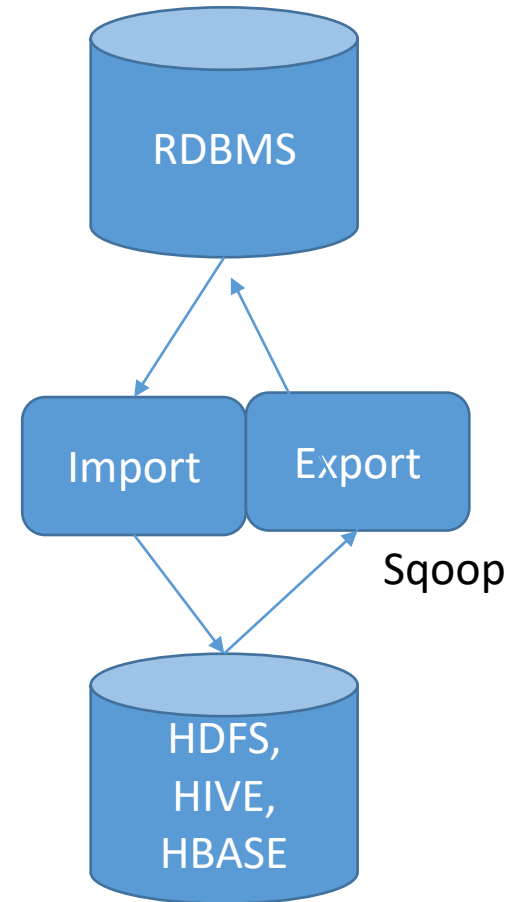
### Why Sqoop ?

- Sometimes we need to use data periodically from a
  - Data warehouse
  - SQL database
- For performing analytics
- And store the data back into an SQL database
- SQOOP → SQL to Hadoop provides this functionality

# BIG DATA

## What is SQOOP?

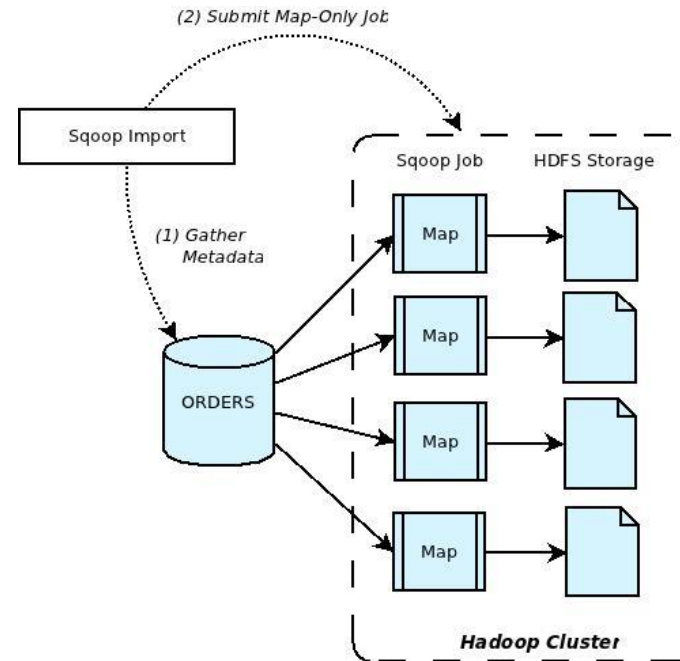
- Bulk Data Transfer Tool – voluminous data
- Import/Export data to/from SQL
  - Defines schema for import
- Integrates with Oozie as an action
- Support plugins for data sources
  - Let's say a newer database that is not supported by default.



# BIG DATA

## How does sqoop work? (IMPORT)

- Step 1
  - Inspects database to gather necessary metadata on data being imported
- Step 2
  - Transfers the data
    - Map only hadoop job
    - Stores data to hdfs directory
    - Imports into csv file, with newline as record separator





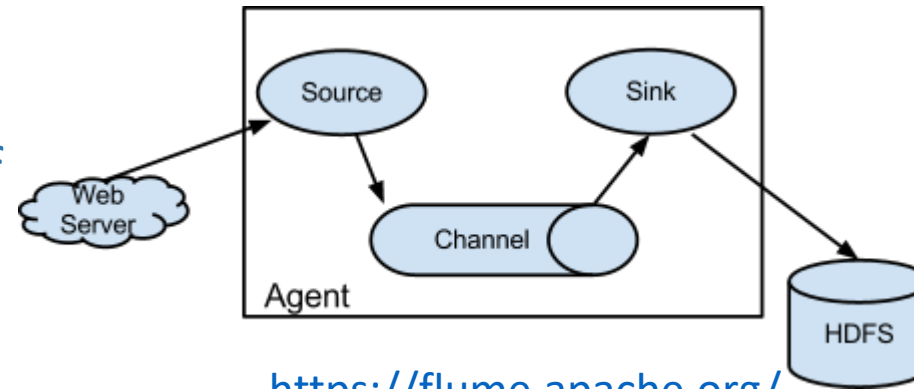


**FLUME**

# BIG DATA

## What is Flume ?

- Meant for collecting large amounts of streaming data
  - Events
  - Logs – like web server logs
- Architecture
  - Sources – accept data from an application
  - Sinks – receive data and store into HDFS
  - Channels – connect sources to sinks
  - Agents run the sources and sinks within Flume



<https://flume.apache.org/>



**THANK YOU**

---

**K V Subramaniam**

Dept. of Computer Science and Engineering

**[subramaniamkv@pes.edu](mailto:subramaniamkv@pes.edu)**