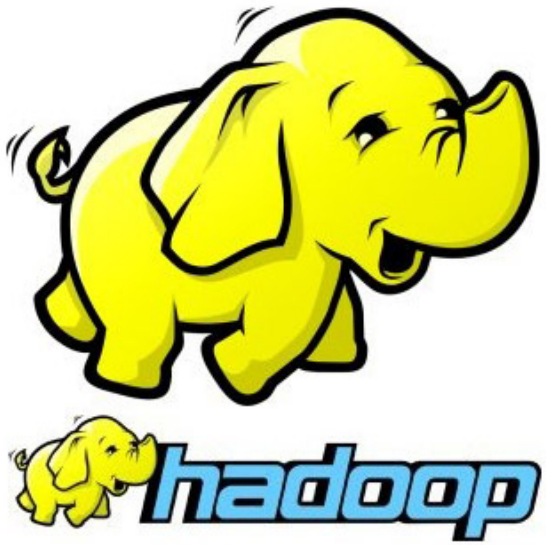
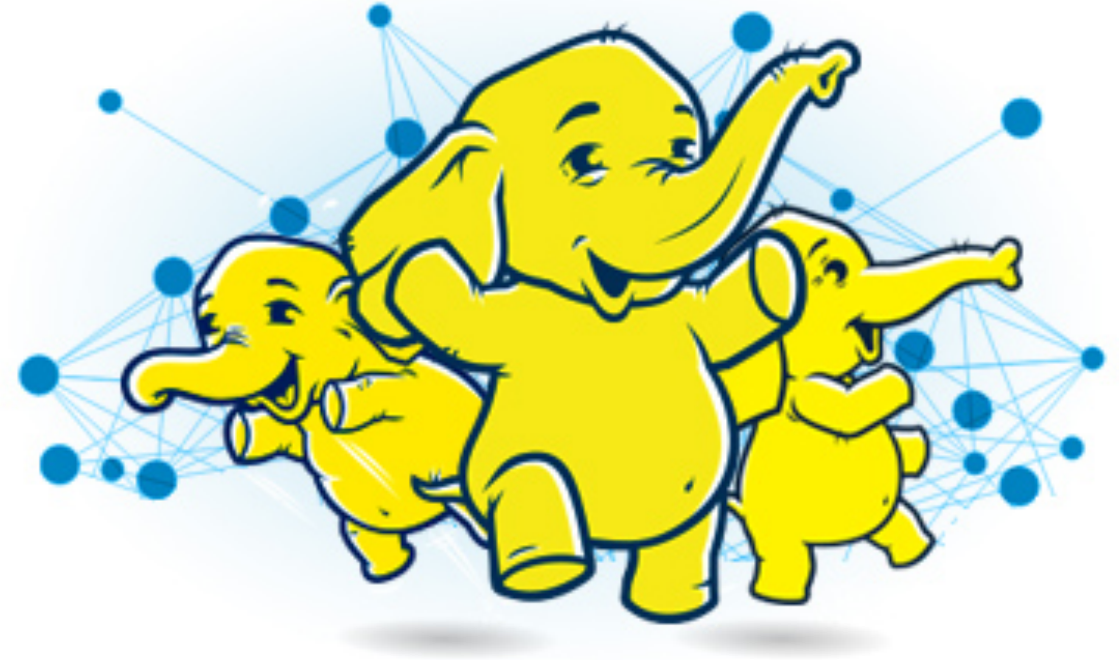


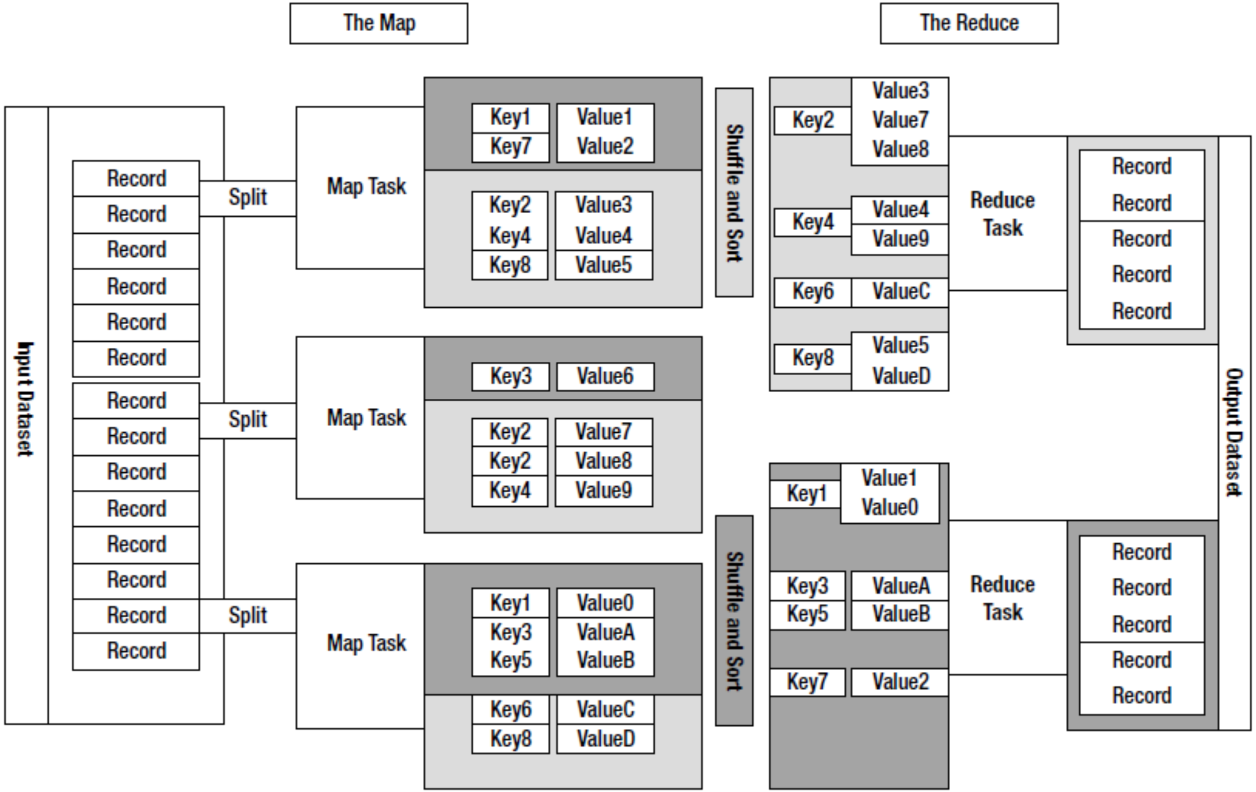
Yaroslav Yermilov

Disclaimer - no words on slides

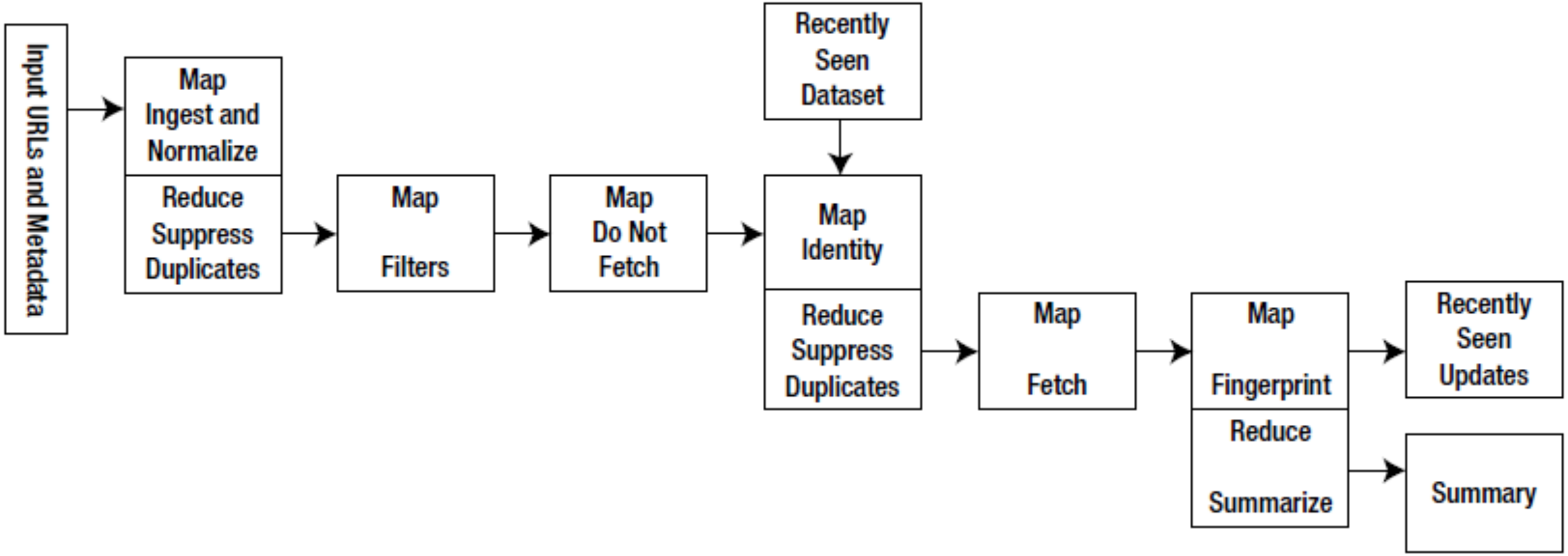


Hadoop MapReduce

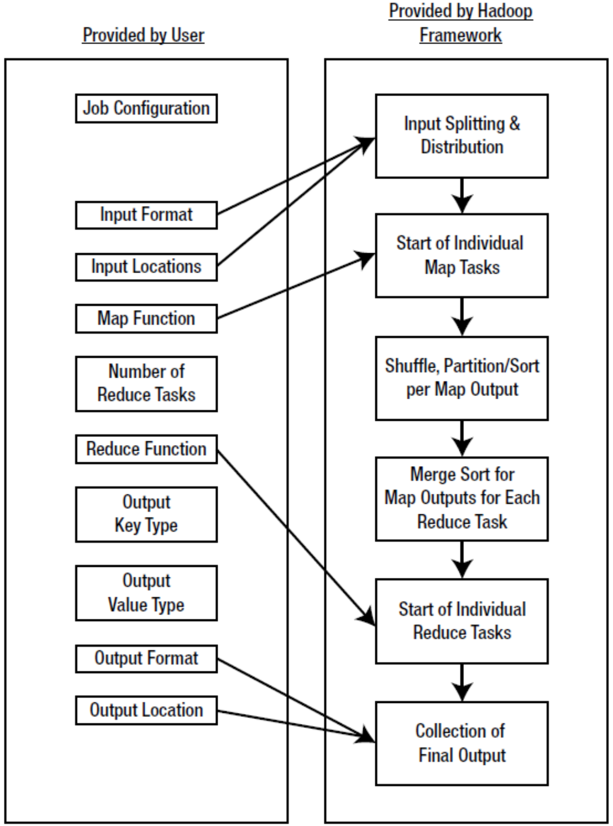
MapReduce model



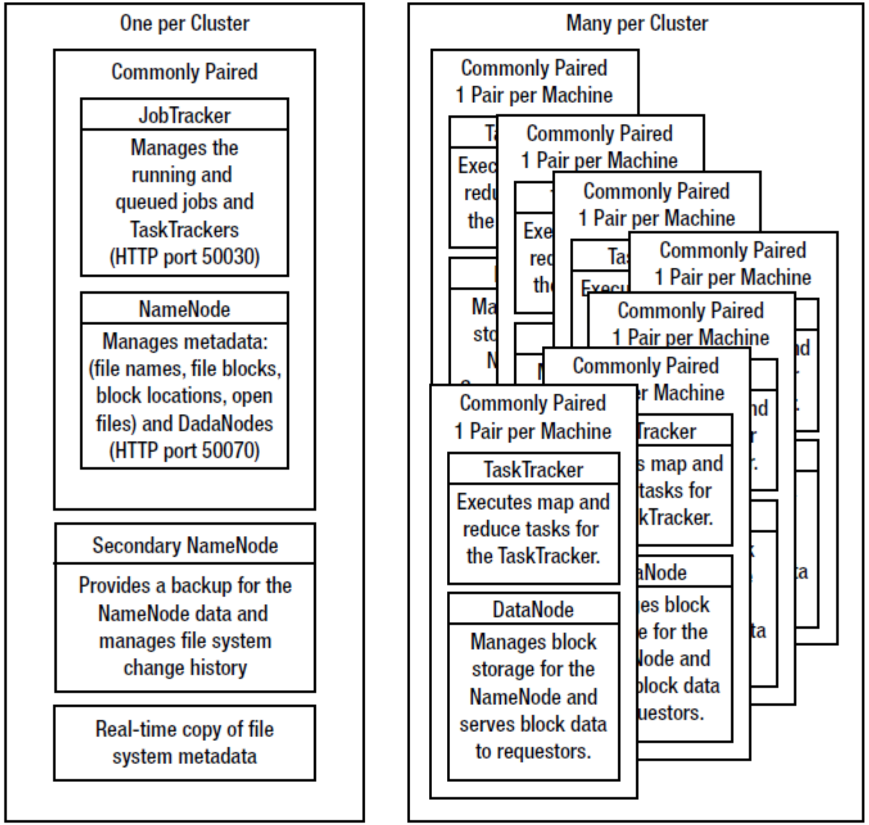
Web crawler example



Parts of MapReduce job


Provided by User
Provided by Hadoop Framework
Job Configuration
Input Format
Input Locations
Map Function
Number of Reduce Tasks
Reduce Function
Output Key Type
Output Value Type
Output Format
Output Location
Input Splitting & Distribution
Start of Individual Map Tasks
Shuffle, Partition/Sort per Map Output
Merge Sort for Map Outputs for Each Reduce Task
Start of Individual Reduce Tasks
Collection of Final Output

Hadoop cluster



One per Cluster

Commonly Paired

JobTracker

Manages the running and queued jobs and TaskTrackers (HTTP port 50030)

NameNode

Manages metadata: (file names, file blocks, block locations, open files) and DadaNodes (HTTP port 50070)

Secondary NameNode

Provides a backup for the NameNode data and manages file system change history

Real-time copy of file system metadata

Many per Cluster

Commonly Paired
1 Pair per Machine

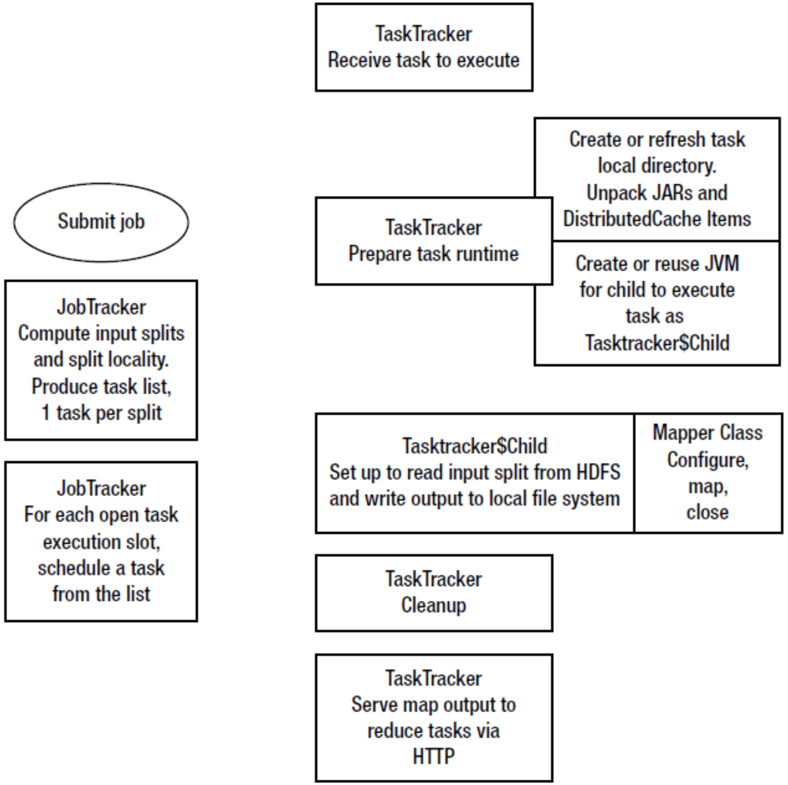
TaskTracker

Executes map and reduce tasks for the TaskTracker.

DataNode

Manages block storage for the NameNode and serves block data to requestors.

How mapper works



Submit job

JobTracker
Compute input splits
and split locality.
Produce task list,
1 task per split

JobTracker
For each open task
execution slot,
schedule a task
from the list

TaskTracker
Receive task to execute

TaskTracker
Prepare task runtime

Create or refresh task
local directory.
Unpack JARs and
DistributedCache Items

Create or reuse JVM
for child to execute
task as
Tasktracker$Child

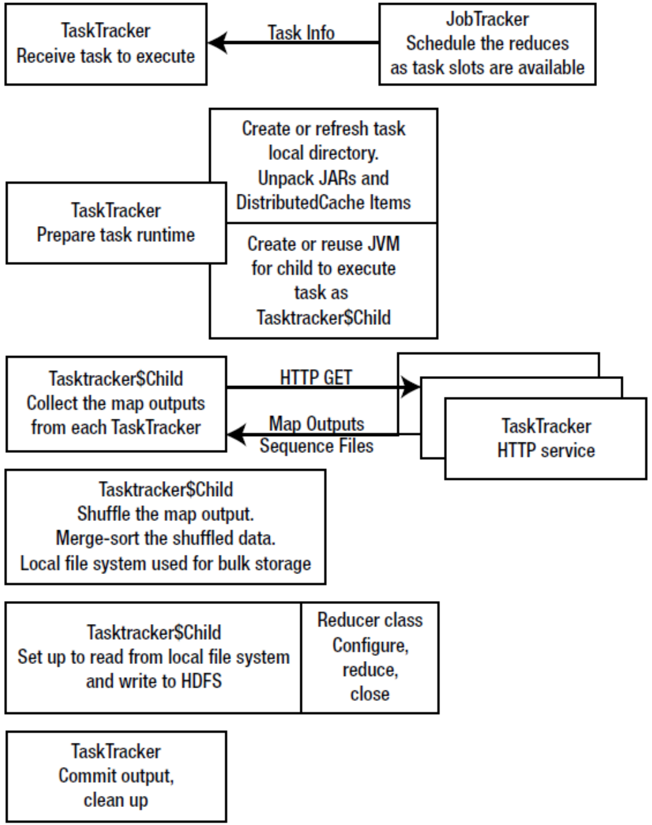
Tasktracker$Child
Set up to read input split from HDFS
and write output to local file system

Mapper Class
Configure,
map,
close

TaskTracker
Cleanup

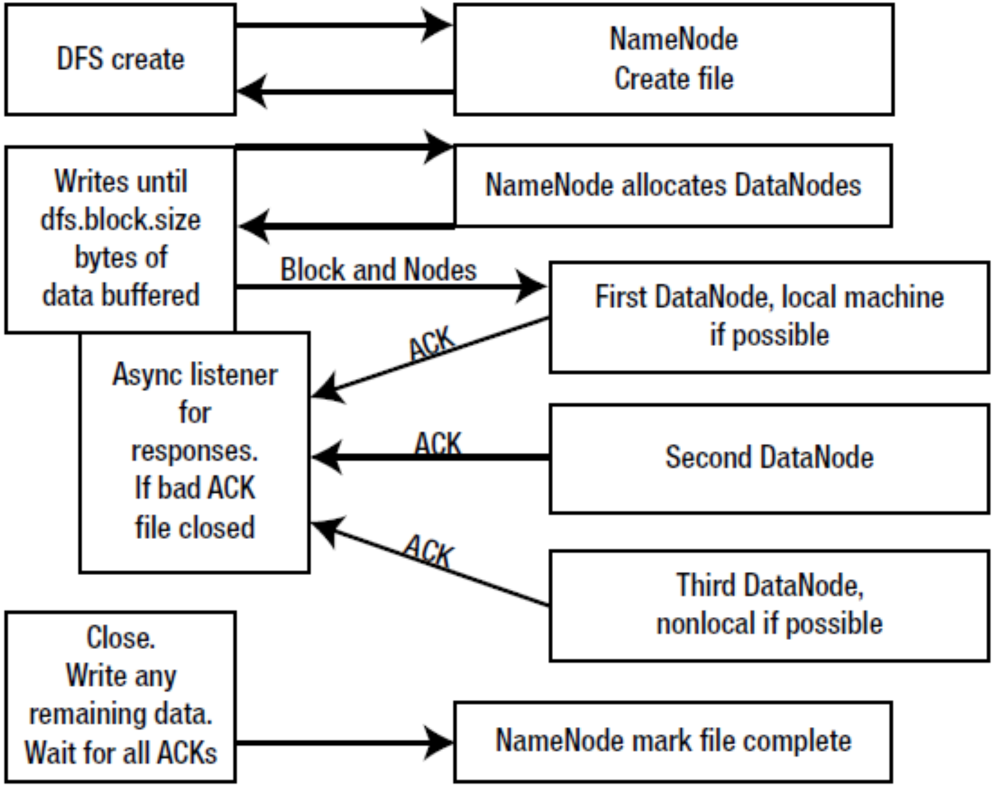
TaskTracker
Serve map output to
reduce tasks via
HTTP

How reducer works



Hadoop HDFS

How HDFS works



DFS create

NameNode
Create file

Writes until
dfs.block.size
bytes of
data buffered

NameNode allocates DataNodes

Block and Nodes

First DataNode, local machine
if possible

ACK

Async listener
for
responses.
If bad ACK
file closed

ACK

Second DataNode

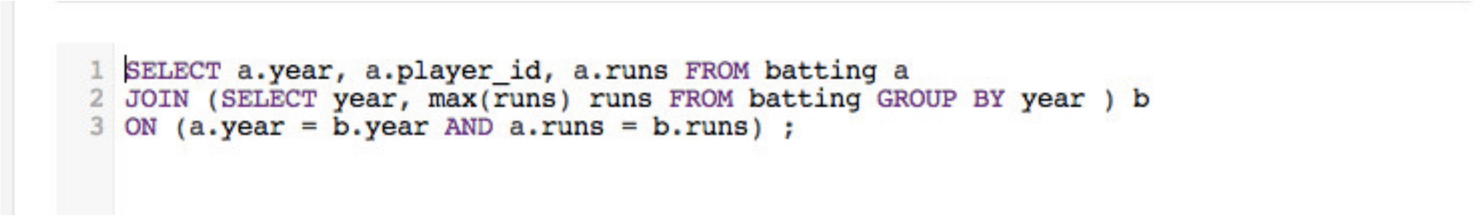
ACK

Third DataNode,
nonlocal if possible

Close.
Write any
remaining data.
Wait for all ACKs

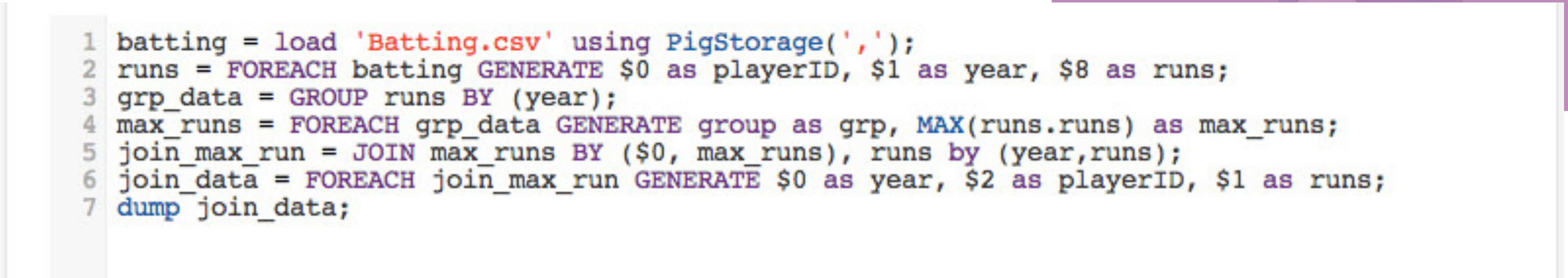
NameNode mark file complete

Hive



```
SELECT a.year, a.player_id, a.runs FROM batting a
JOIN (SELECT year, max(runs) runs FROM batting GROUP BY year ) b
ON (a.year = b.year AND a.runs = b.runs) ;
```

Pig



```
batting = load 'Batting.csv' using PigStorage(',');
runs = FOREACH batting GENERATE $0 as playerID, $1 as year, $8 as runs;
grp_data = GROUP runs BY (year);
max_runs = FOREACH grp_data GENERATE group as grp, MAX(runs.runs) as max_runs;
join_max_run = JOIN max_runs BY ($0, max_runs), runs by (year,runs);
join_data = FOREACH join_max_run GENERATE $0 as year, $2 as playerID, $1 as runs;
dump join_data;
```

HBase

Sqoop

Flume

Zookeeper

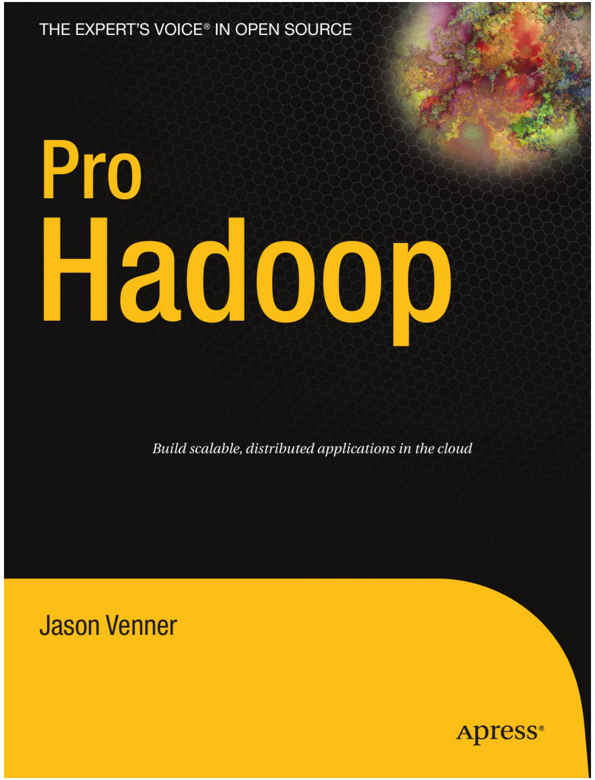
Oozie

Falcon

Mahout

Avro

Sources



THE EXPERT'S VOICE® IN OPEN SOURCE

Pro Hadoop

Build scalable, distributed applications in the cloud

Jason Venner

Apress®