Hadoop Monitoring, SOLR vs Elastic Search

- Paolo Nesi, Gianni Pantaleo, Imad Zaza

- **DISIT Lab**
- Dipartimento di Ingegneria dell'Informazione, DINFO
- Università degli Studi di Firenze
  Via S. Marta 3, 50139, Firenze, Italy
  Tel: +39-055-4796567, fax: +39-055-4796363
- **http://www.disit.dinfo.unifi.it** ***alias*** http://www.disit.org

Ottobre 2020

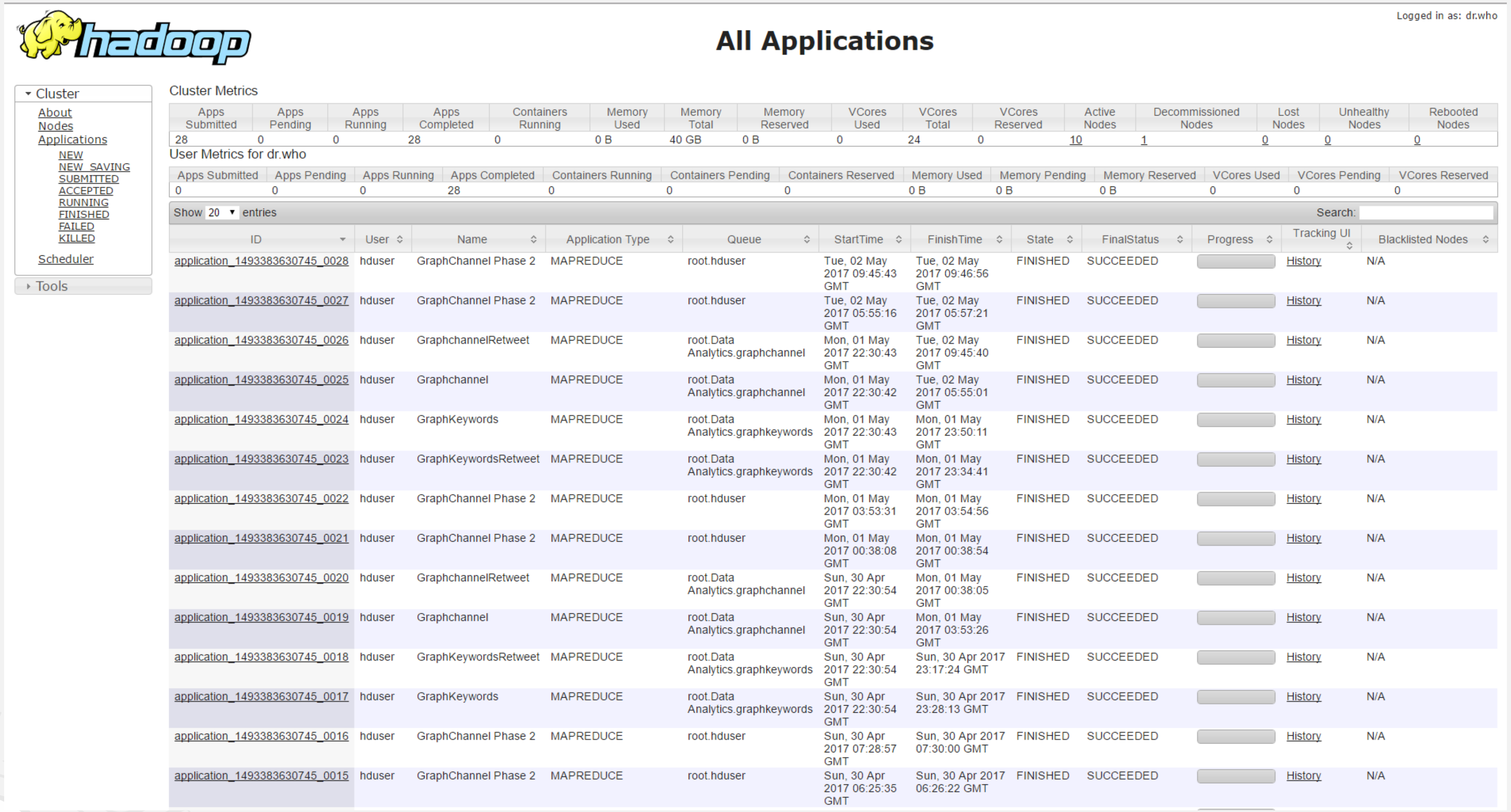
Agenda

- Monitoring
- Apache HBASE
- Apache Phoenix
- Case studio
- Apache Solr
- Elastic Search

What really going on !?

- How many maps are issued?
- Where are computed the maps?
- Does my job executing normally or something is going wrong !?
- Does my cluster being underutilzied or overutilized !?

Basic web interface



Logged in as: dr.who

All Applications

- Cluster
  - About
  - Nodes
  - Applications
    - NEW
    - NEW_SAVING
    - SUBMITTED
    - ACCEPTED
    - RUNNING
    - FINISHED
    - FAILED
    - KILLED
  - Scheduler
- Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
28	0	0	28	0	0 B	40 GB	0 B	0	24	0	10	1	0	0	0

User Metrics for dr.who

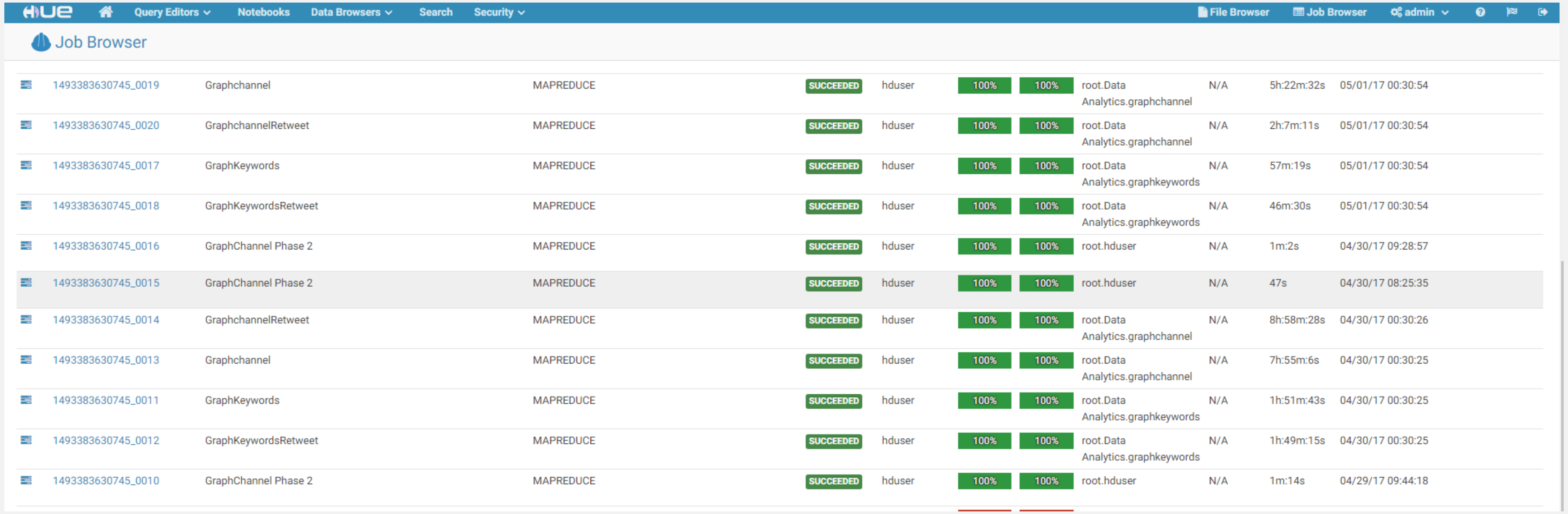
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	VCores Used	VCores Pending	VCores Reserved
0	0	0	28	0	0	0	0 B	0 B	0 B	0	0	0

Show 20 ▾ entries

Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
application_1493383630745_0028	hduser	GraphChannel Phase 2	MAPREDUCE	root.hduser	Tue, 02 May 2017 09:45:43 GMT	Tue, 02 May 2017 09:46:56 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0027	hduser	GraphChannel Phase 2	MAPREDUCE	root.hduser	Tue, 02 May 2017 05:55:16 GMT	Tue, 02 May 2017 05:57:21 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0026	hduser	GraphchannelRetweet	MAPREDUCE	root.Data Analytics.graphchannel	Mon, 01 May 2017 22:30:43 GMT	Tue, 02 May 2017 09:45:40 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0025	hduser	Graphchannel	MAPREDUCE	root.Data Analytics.graphchannel	Mon, 01 May 2017 22:30:42 GMT	Tue, 02 May 2017 05:55:01 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0024	hduser	GraphKeywords	MAPREDUCE	root.Data Analytics.graphkeywords	Mon, 01 May 2017 22:30:43 GMT	Mon, 01 May 2017 23:50:11 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0023	hduser	GraphKeywordsRetweet	MAPREDUCE	root.Data Analytics.graphkeywords	Mon, 01 May 2017 22:30:42 GMT	Mon, 01 May 2017 23:34:41 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0022	hduser	GraphChannel Phase 2	MAPREDUCE	root.hduser	Mon, 01 May 2017 03:53:31 GMT	Mon, 01 May 2017 03:54:56 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0021	hduser	GraphChannel Phase 2	MAPREDUCE	root.hduser	Mon, 01 May 2017 00:38:08 GMT	Mon, 01 May 2017 00:38:54 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0020	hduser	GraphchannelRetweet	MAPREDUCE	root.Data Analytics.graphchannel	Sun, 30 Apr 2017 22:30:54 GMT	Mon, 01 May 2017 00:38:05 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0019	hduser	Graphchannel	MAPREDUCE	root.Data Analytics.graphchannel	Sun, 30 Apr 2017 22:30:54 GMT	Mon, 01 May 2017 03:53:26 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0018	hduser	GraphKeywordsRetweet	MAPREDUCE	root.Data Analytics.graphkeywords	Sun, 30 Apr 2017 22:30:54 GMT	Sun, 30 Apr 2017 23:17:24 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0017	hduser	GraphKeywords	MAPREDUCE	root.Data Analytics.graphkeywords	Sun, 30 Apr 2017 22:30:54 GMT	Sun, 30 Apr 2017 23:28:13 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0016	hduser	GraphChannel Phase 2	MAPREDUCE	root.hduser	Sun, 30 Apr 2017 07:28:57 GMT	Sun, 30 Apr 2017 07:30:00 GMT	FINISHED	SUCCEEDED		History	N/A
application_1493383630745_0015	hduser	GraphChannel Phase 2	MAPREDUCE	root.hduser	Sun, 30 Apr 2017 06:25:35 GMT	Sun, 30 Apr 2017 06:26:22 GMT	FINISHED	SUCCEEDED		History	N/A

Advanced web interface



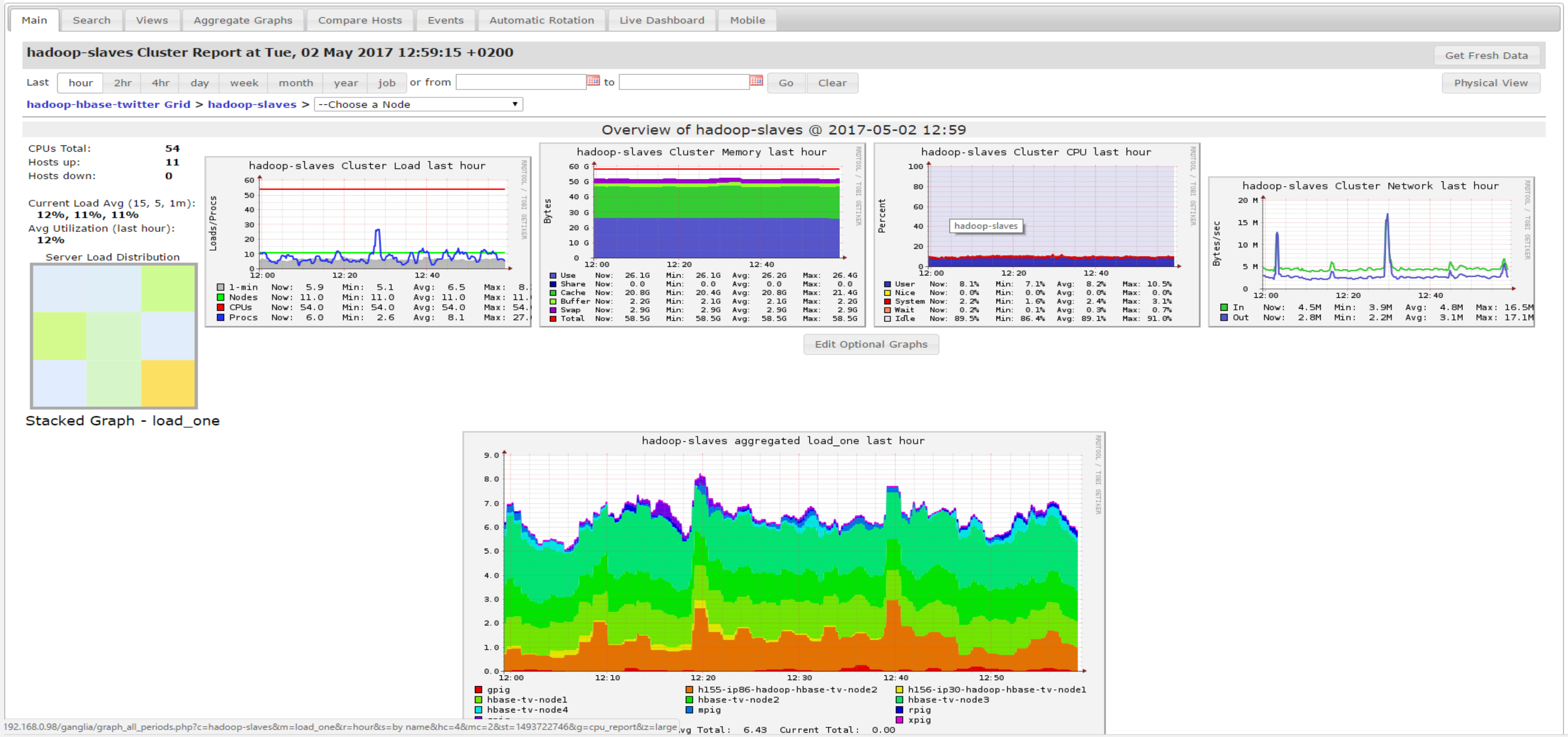
Job Browser

1493383630745_0019	Graphchannel	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphchannel	N/A	5h:22m:32s	05/01/17 00:30:54
1493383630745_0020	GraphchannelRetweet	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphchannel	N/A	2h:7m:11s	05/01/17 00:30:54
1493383630745_0017	GraphKeywords	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphkeywords	N/A	57m:19s	05/01/17 00:30:54
1493383630745_0018	GraphKeywordsRetweet	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphkeywords	N/A	46m:30s	05/01/17 00:30:54
1493383630745_0016	GraphChannel Phase 2	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.hduser	N/A	1m:2s	04/30/17 09:28:57
1493383630745_0015	GraphChannel Phase 2	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.hduser	N/A	47s	04/30/17 08:25:35
1493383630745_0014	GraphchannelRetweet	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphchannel	N/A	8h:58m:28s	04/30/17 00:30:26
1493383630745_0013	Graphchannel	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphchannel	N/A	7h:55m:6s	04/30/17 00:30:25
1493383630745_0011	GraphKeywords	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphkeywords	N/A	1h:51m:43s	04/30/17 00:30:25
1493383630745_0012	GraphKeywordsRetweet	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.Data Analytics.graphkeywords	N/A	1h:49m:15s	04/30/17 00:30:25
1493383630745_0010	GraphChannel Phase 2	MAPREDUCE	SUCCEEDED	hduser	100%	100%	root.hduser	N/A	1m:14s	04/29/17 09:44:18

What really really going on !?

- The aforementioned application doesn’t give specific information …
- Unix tools
- Batch System tools
- Really need something that can provide a quick visual overview of the health and load on your cluster

Ganglia



hadoop-slaves Cluster Report at Tue, 02 May 2017 12:59:15 +0200

hadoop-hbase-twitter Grid > hadoop-slaves >

Overview of hadoop-slaves @ 2017-05-02 12:59

CPUs Total: 54
Hosts up: 11
Hosts down: 0

Current Load Avg (15, 5, 1m):
12%, 11%, 11%
Avg Utilization (last hour):
12%

Server Load Distribution

hadoop-slaves Cluster Load last hour

```
1-min   Now:   5.9   Min:   5.1   Avg:   6.5   Max:   8.
Nodes   Now:  11.0   Min:  11.0   Avg:  11.0   Max:  11.
CPUs    Now:  54.0   Min:  54.0   Avg:  54.0   Max:  54.
Procs   Now:   6.0   Min:   2.6   Avg:   8.1   Max:  27.
```

hadoop-slaves Cluster Memory last hour

```
Use     Now:   26.1G   Min:   26.1G   Avg:   26.2G   Max:   26.4G
Share   Now:    0.0    Min:    0.0    Avg:    0.0    Max:    0.0
Cache   Now:   20.8G   Min:   20.4G   Avg:   20.8G   Max:   21.4G
Buffer  Now:    2.2G   Min:    2.1G   Avg:    2.1G   Max:    2.2G
Swap    Now:    2.9G   Min:    2.9G   Avg:    2.9G   Max:    2.9G
Total   Now:   58.5G   Min:   58.5G   Avg:   58.5G   Max:   58.5G
```

hadoop-slaves Cluster CPU last hour

```
User    Now:   8.1%   Min:   7.1%   Avg:   8.2%   Max:  10.5%
Nice    Now:   0.0%   Min:   0.0%   Avg:   0.0%   Max:   0.0%
System  Now:   2.2%   Min:   1.6%   Avg:   2.4%   Max:   3.1%
Wait    Now:   0.2%   Min:   0.1%   Avg:   0.3%   Max:   0.7%
Idle    Now:  89.5%   Min:  86.4%   Avg:  89.1%   Max:  91.0%
```

hadoop-slaves Cluster Network last hour

```
In    Now:   4.5M   Min:   3.9M   Avg:   4.8M   Max:  16.5M
Out   Now:   2.8M   Min:   2.2M   Avg:   3.1M   Max:  17.1M
```

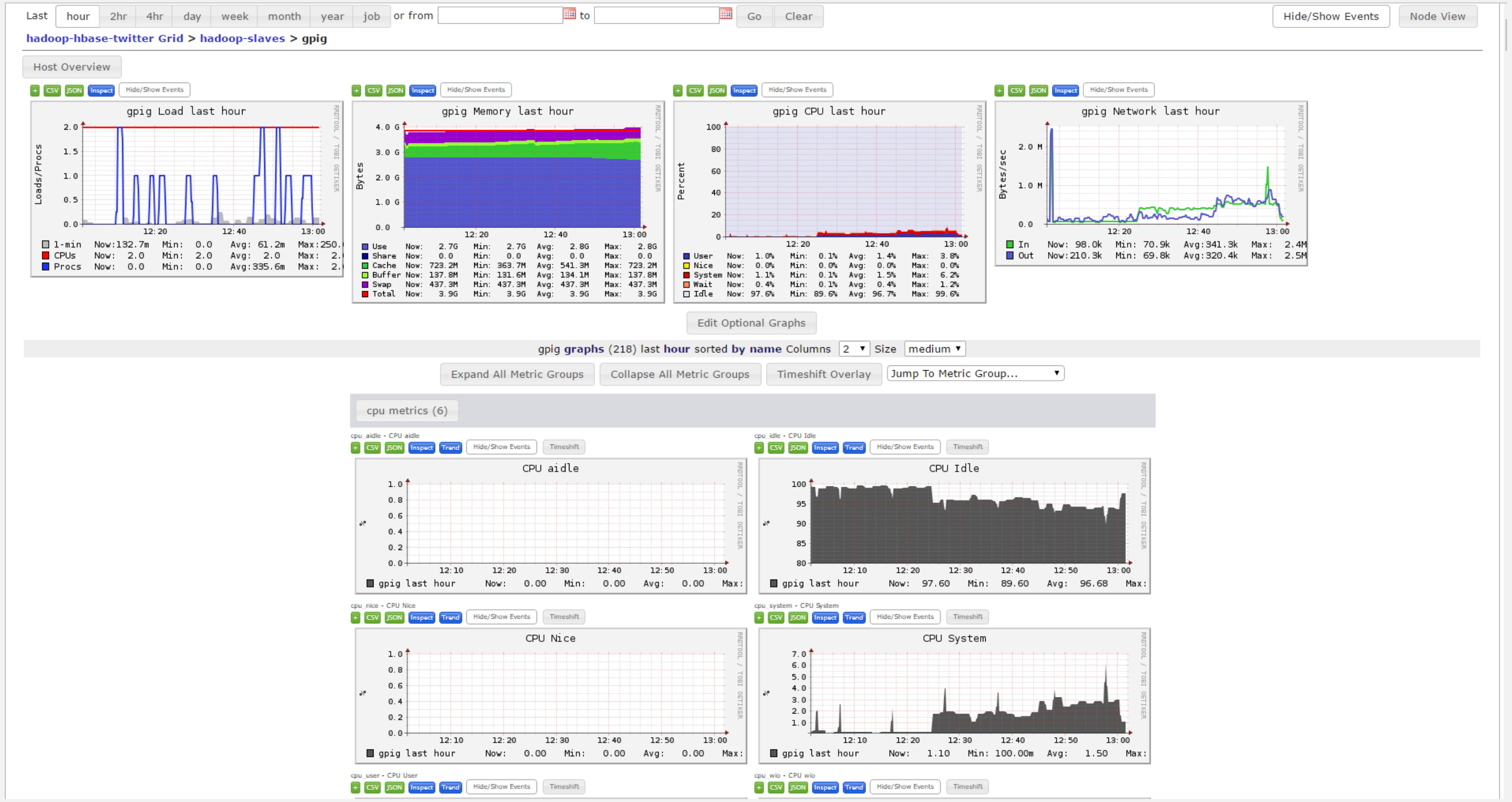
Stacked Graph - load_one

hadoop-slaves aggregated load_one last hour

```
gpig                 h155-ip86-hadoop-hbase-tv-node2   h156-ip30-hadoop-hbase-tv-node1
hbase-tv-node1       hbase-tv-node2                    hbase-tv-node3
hbase-tv-node4       mpig                              rpig
                                                       xpig
vg Total:   6.43   Current Total:   0.00
```

192.168.0.98/ganglia/graph_all_periods.php?c=hadoop-slaves&m=load_one&r=hour&s=by name&hc=4&mc=2&st=1493722746&g=cpu_report&z=large

Host level detail

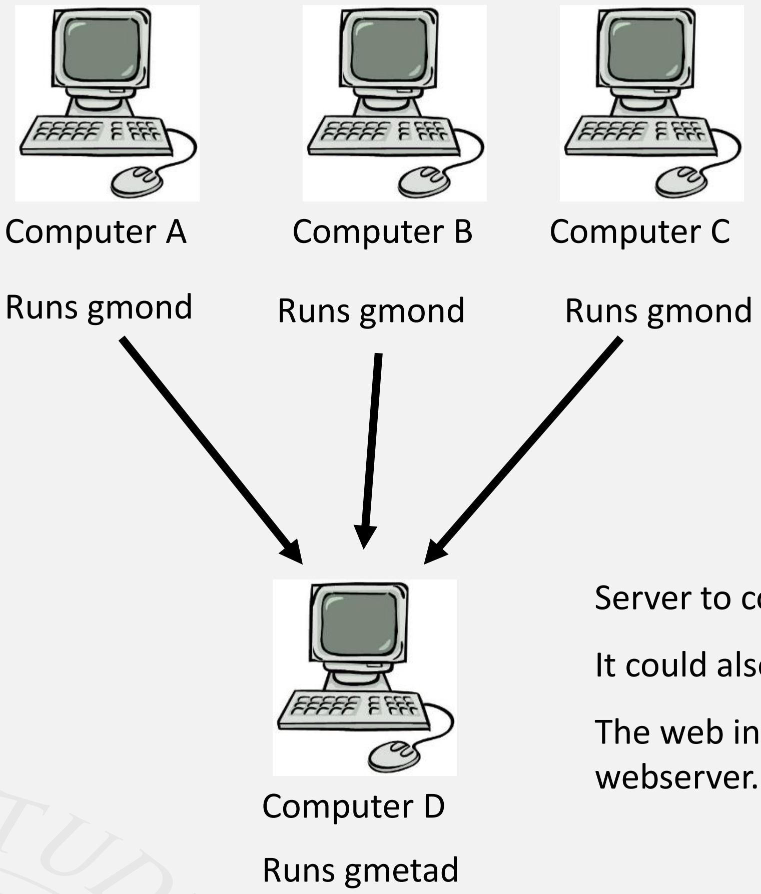


How does Ganglia work?

- Ganglia works through a small agent, *gmond*, on each node or machine to be monitored. You can distribute a single gmond instance to lots of machines at once. Gmonds communicate the state of their local node to a machine running a Master *gmetad* instance.
- The server uses RRDtool to store the data over time
- The Ganglia framework can be extended to monitor many parameters.

Setup

The software can be downloaded from http://ganglia.sourceforge.net/

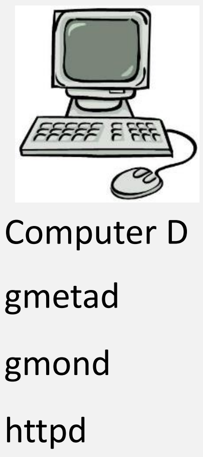


Clients just have to run gmond, which is configured by /etc/gmond.conf

Server to collect the data runs gmetad.

It could also run gmond to monitor itself.

The web interface needs to run on a webserver.



Client Setup

Computer A

Runs gmond

Computer B

Runs gmond

Computer C

Runs gmond

Apt-get install ganglia-gmond

edit config file

service gmond start

chkconfig gmond on

/etc/gmond.conf extracts

```
cluster {
  name = "LCG Workers"
}
/* Feel free to specify as many udp_send_channels as you like.  Gmond
   used to only support having a single channel */
udp_send_channel {
  mcast_join = 239.2.11.95
  port = 8649
}
```

```
udp_send_channel {
  port = 8649
  host = pplxconfig
}
```

```
/* You can specify as many udp_recv_channels as you like as well. */
udp_recv_channel {
  mcast_join = 239.2.11.95
  port = 8649
  bind = 239.2.11.95
}

/* You can specify as many tcp_accept_channels as you like to share
   an xml description of the state of the cluster */
tcp_accept_channel {
  port = 8649
}
```

Server Setup

Computer D

gmetad

gmond

httpd

Extracts from /etc/gmetad.conf

```
data_source "LCG Workers" 127.0.0.1:8655 computerA.physics.ox.ac.uk
ComputerB.physics.ox.ac.uk computerC.physics.ox.ac.uk:8655

data_source "LCG Servers" t2se01.physics.ox.ac.uk:8656
t2ce02.physics.ox.ac.uk:8656 gridlogger.physics.ox.ac.uk:8656
```

Aptitude install ganglia-gmond ganglia-gmetad ganglia-web

edit /etc/gmond.conf

edit /etc/gmetad.conf

Agenda

- Monitoring
- Apache HBASE
- Case studio
- Apache Phoenix
- Apache Solr
- Elastic Search

Introduction

- Hadoop is a framework that supports operations on a large amount of data.
- Hadoop includes the Hadoop Distributed File System (HDFS)
- HDFS does a good job of storing large amounts of data, but lacks quick random read/write capability.

Introduction (cont.)

- We need a tabular form to store our data
- The storing system must preserve the advantages of hadoop
  - Fault tolerance
  - High perfomance on large amount of data

- Limits of Relational Databases (RDBMS)
- OLTP vs OLAP
- Big Data storing paradigms…

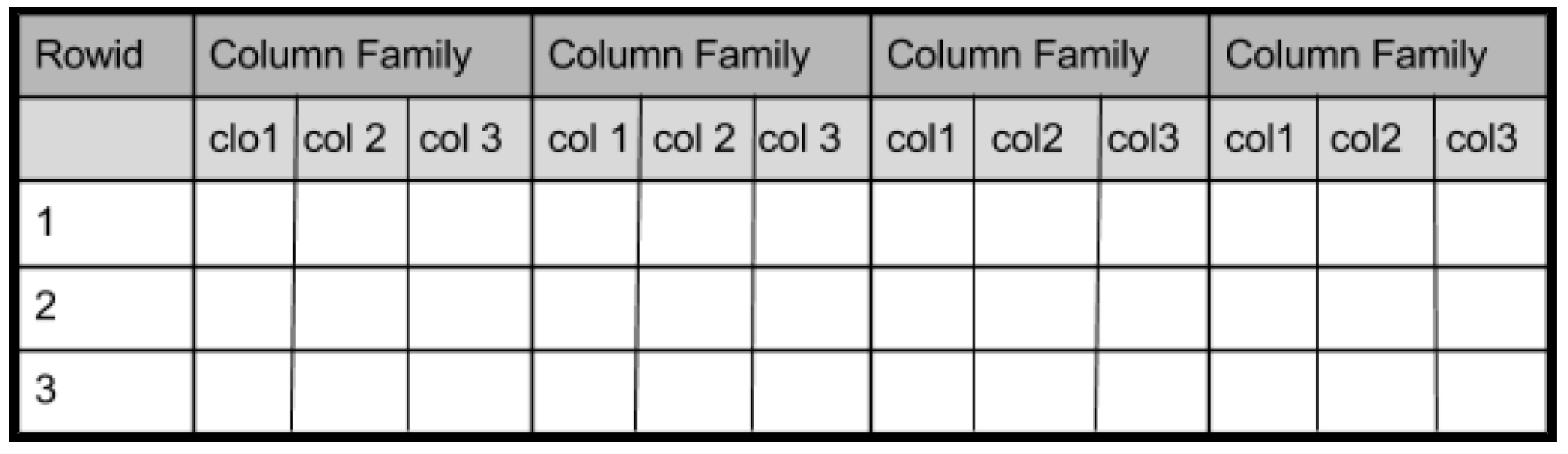
Introduction (cont.)

- HBase is a NoSQL (schema-less), column-oriented database.
- It is open source, sparse, consistently distributed, sorted map modeled after Google’s BigTable.
- Developed as part of Apache’s Hadoop project and runs on top of Hadoop Distributed File System.
- Horizontaly, linearly scalable

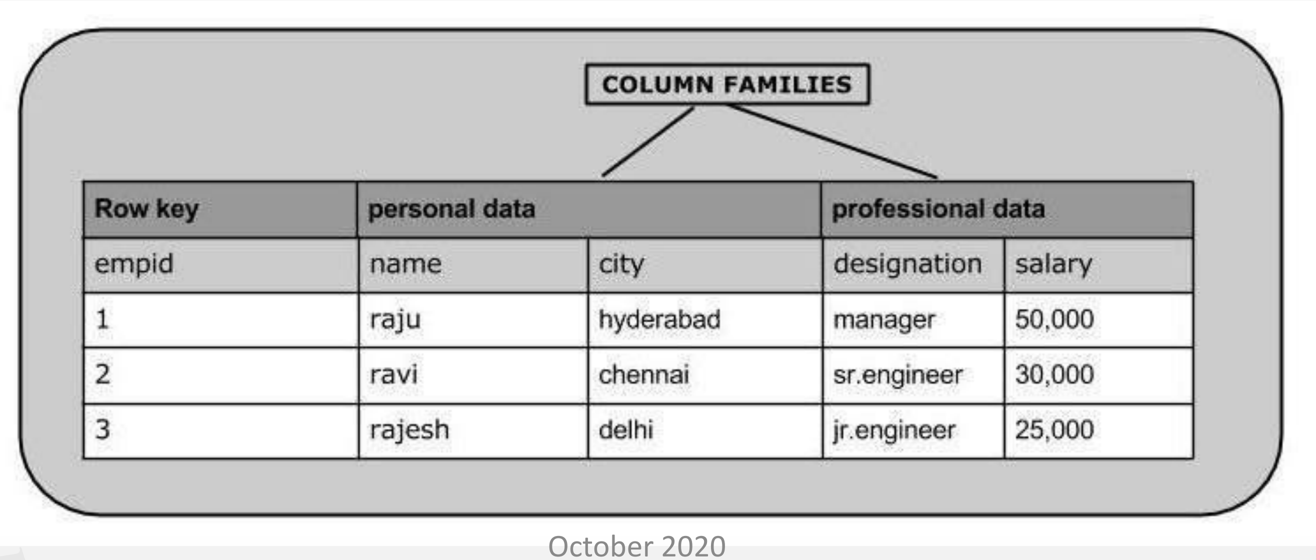
Conceptual View

- A data row has a sortable row key
- Table is a collection of rows.
- Row is a collection of column families.
- Column family is a collection of columns.
- Column is a collection of key value pairs.
- A Time Stamp is designated automatically if not artificially.
- *<family>:<identifier>*

Conceptual View



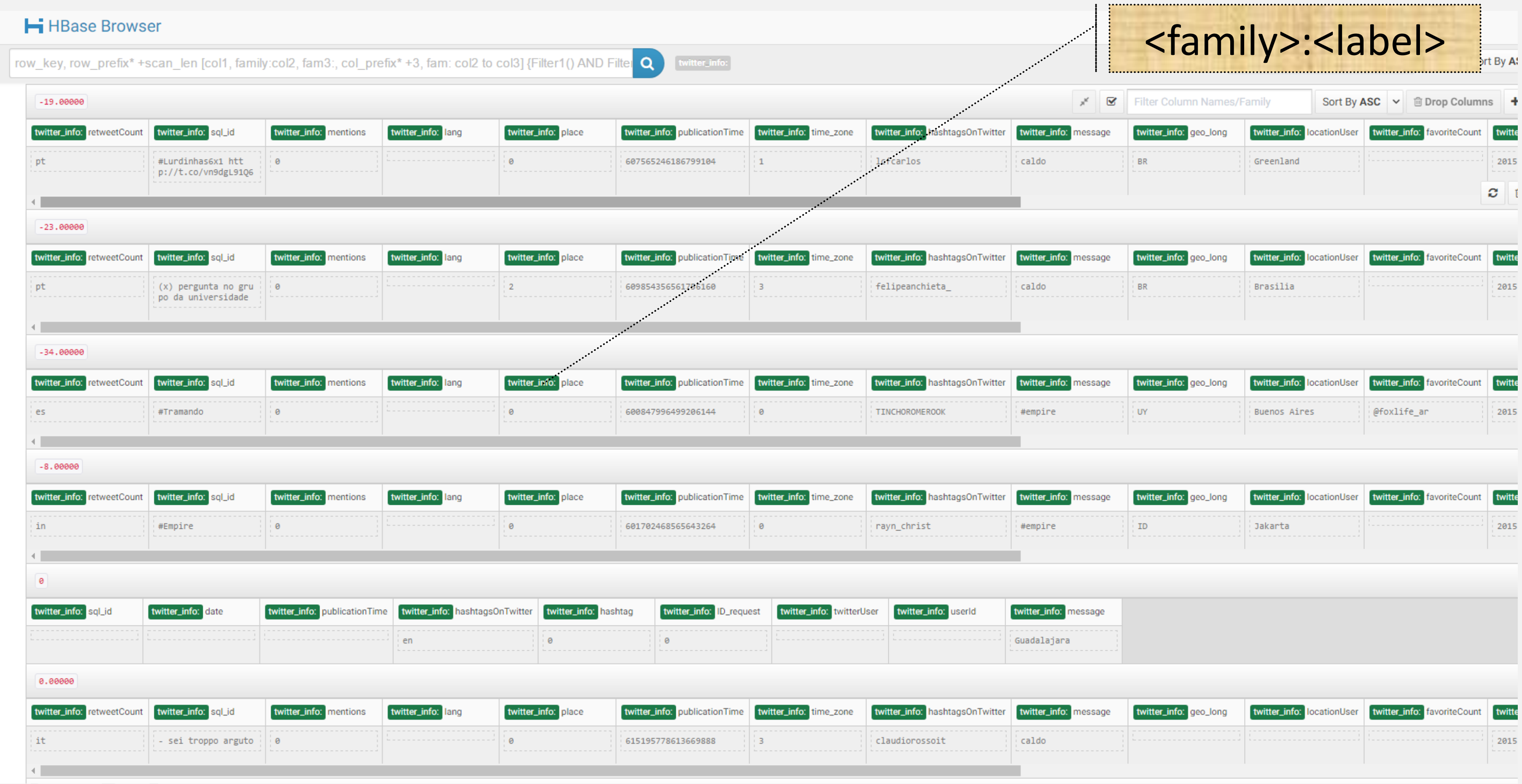
Rowid	Column Family			Column Family			Column Family			Column Family		
	clo1	col 2	col 3	col 1	col 2	col 3	col1	col2	col3	col1	col2	col3
1												
2												
3												



COLUMN FAMILIES

Row key	personal data		professional data	
empid	name	city	designation	salary
1	raju	hyderabad	manager	50,000
2	ravi	chennai	sr.engineer	30,000
3	rajesh	delhi	jr.engineer	25,000

Hbase table


HBase Browser
row_key, row_prefix* +scan_len [col1, family:col2, fam3:, col_prefix* +3, fam: col2 to col3] {Filter1() AND Filte
twitter_info:
Filter Column Names/Family
Sort By ASC
Drop Columns
rt By A

<family>:<label>

-19.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
pt	#Lurdinhas6x1 http://t.co/vn9dgL91Q6	0		0	607565246186799104	1	lafcarlos	caldo	BR	Greenland		2015

-23.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
pt	(x) pergunta no grupo da universidade	0		2	609854356561736160	3	felipeanchieta_	caldo	BR	Brasilia		2015

-34.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
es	#Tramando	0		0	600847996499206144	0	TINCHOROMEROOK	#empire	UY	Buenos Aires	@foxlife_ar	2015

-8.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
in	#Empire	0		0	601702468565643264	0	rayn_christ	#empire	ID	Jakarta		2015

0

twitter_info: sql_id	twitter_info: date	twitter_info: publicationTime	twitter_info: hashtagsOnTwitter	twitter_info: hashtag	twitter_info: ID_request	twitter_info: twitterUser	twitter_info: userId	twitter_info: message
			en	0	0			Guadalajara

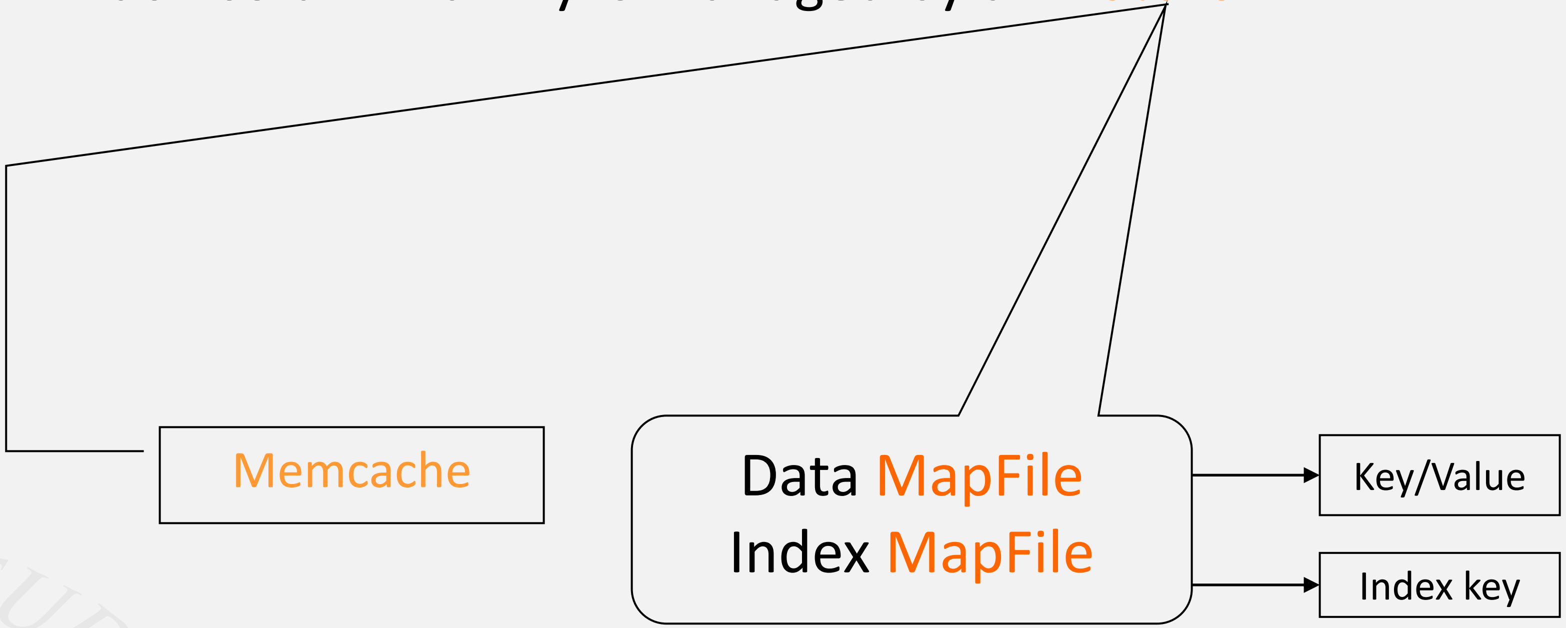
0.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
it	- sei troppo arguto	0		0	615195778613669888	3	claudiorossoit	caldo				2015

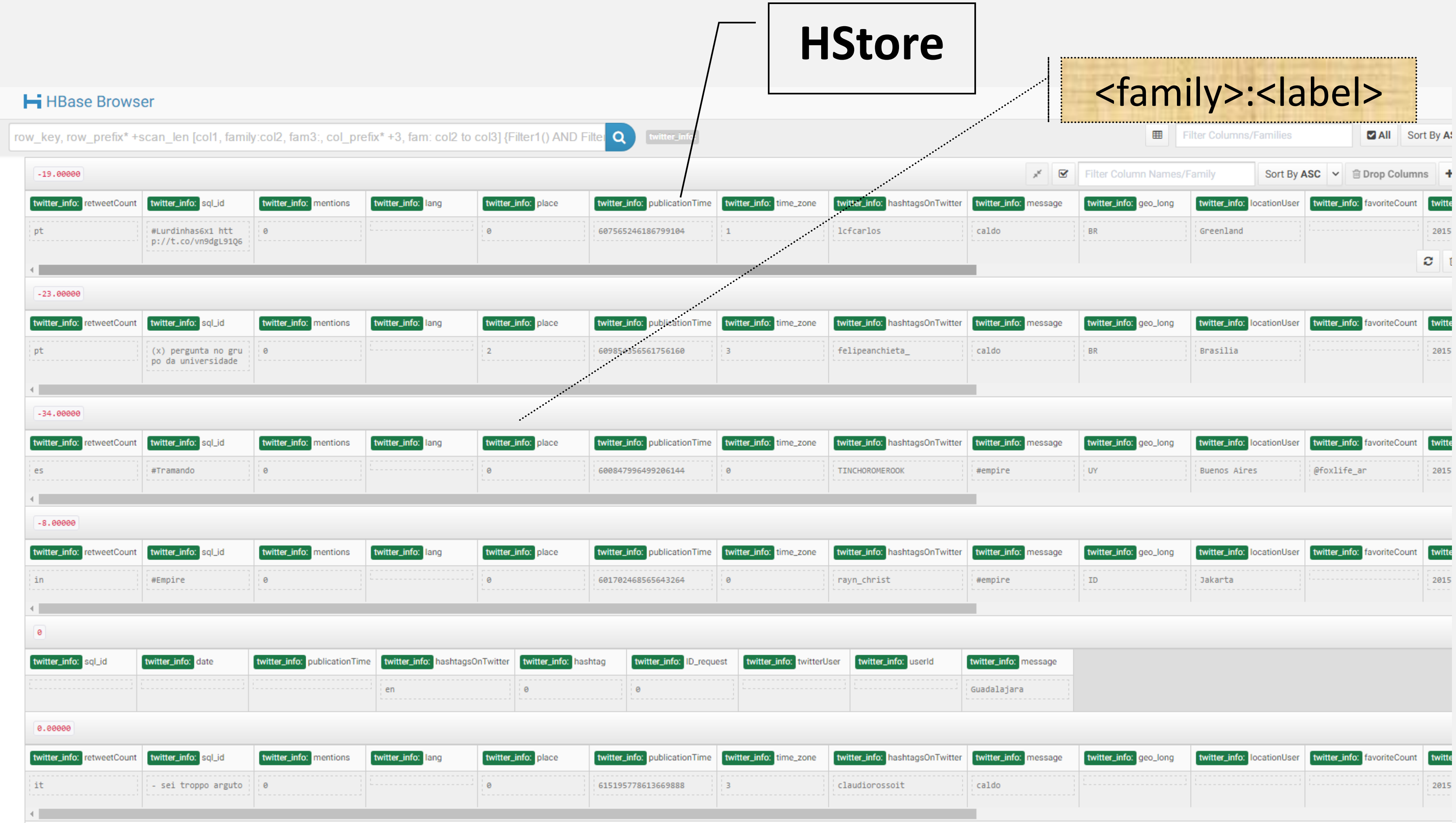
Let’s try it !

Physical Storage View

- Physically, tables are stored on a per-column family basis.
- Empty cells are not stored in a column-oriented storage format.
- Each column family is managed by an HStore.



Hbase table



HStore

<family>:<label>

HBase Browser

row_key, row_prefix* +scan_len [col1, family:col2, fam3:, col_prefix* +3, fam: col2 to col3] {Filter1() AND Filte

-19.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
pt	#Lurdinhas6x1 http://t.co/vn9dgL91Q6	0		0	607565246186799104	1	lcfcarlos	caldo	BR	Greenland		2015

-23.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
pt	(x) pergunta no grupo da universidade	0		2	609854356561756160	3	felipeanchieta_	caldo	BR	Brasilia		2015

-34.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
es	#Tramando	0		0	600847996499206144	0	TINCHOROMEROOK	#empire	UY	Buenos Aires	@foxlife_ar	2015

-8.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
in	#Empire	0		0	601702468565643264	0	rayn_christ	#empire	ID	Jakarta		2015

0

twitter_info: sql_id	twitter_info: date	twitter_info: publicationTime	twitter_info: hashtagsOnTwitter	twitter_info: hashtag	twitter_info: ID_request	twitter_info: twitterUser	twitter_info: userId	twitter_info: message
			en	0	0			Guadalajara

0.00000

twitter_info: retweetCount	twitter_info: sql_id	twitter_info: mentions	twitter_info: lang	twitter_info: place	twitter_info: publicationTime	twitter_info: time_zone	twitter_info: hashtagsOnTwitter	twitter_info: message	twitter_info: geo_long	twitter_info: locationUser	twitter_info: favoriteCount	twitte
it	- sei troppo arguto	0		0	615195778613669888	3	claudiorossoit	caldo				2015

Physical Storage View

- Each cell value of the table has a timestamp
- Subsequent column values are stored contiguously on the disk.

What we can do ?

- Mysql-like commands
  - Create table
  - Drop table
  - Insert data in table
  - Delete data
  - Query data
- Mysql-like interface (cli, API)

```
$ hbase shell
```

Hbase commands

- Table manipulations
  - create ‘t1’, ‘f1’, ‘f2’, ‘f3’

    create ‘<table-name>’, ’<column-family1>’, ’<column-family2>’ ……

  - alter 'tablename', NAME => ’<new-column-family>’
  - Disable/enable ‘t1’

    enable/disable ‘<table-name>’

  - drop ‘t1’ (but t1 must be disabled)

    drop ‘<table-name>’

More commands

- put ‘t1’, ‘r1’, ‘c1’, ‘value1’, ts1

  put ‘<table-name>’, ’row-key’, ’columnfamily:columnname’, ’value’, ’timestamp’

- delete ‘t1’, ‘r1’, ‘c1’
- delete ‘t1’, ‘r1’, ‘c1’,’ts1’
- get ‘t1’, ‘r1’ → scan ‘t1’
- get ‘t1’, ‘r1’, {TIMERANGE => [ts1, ts2]} → scan ‘t1’, {TIMERANGE => [ts1, ts2]}
- get ‘t1’, ‘r1’, {COLUMN => ‘c1’} → scan ‘t1’, {COLUMN => ‘c1’}

More commands

- scan 'syslog', {COLUMNS => ['msg:body', 'msg:timestamp'], LIMIT => 100}
- scan 'syslog', {COLUMN => 'msg:body', FILTER => "ValueFilter(=, 'binaryprefix:<*STRING_TO_MATCH*>’)”}
- scan 'syslog', {COLUMN => 'msg:body', FILTER => "ValueFilter(=,'regexstring:.*prova.*’)”}

Trace

- create ’students’, cf=[’a’,’b’]
- put ’students’, ’pippo’, ’a:name’, ‘pippo’
- put ’students’, ‘pippo’, ’a:age’, 20
- put ’students’, ’pluto, ’a:name’, ’pluto’
- scan ’students’

HBASE Client

- Cli : hbase
- Java API
  - HTable hTable = new HTable(config, "students");
  - Put p = new Put(Bytes.toBytes("row1"));
  - p.add(Bytes.toBytes("personal"),Bytes.toBytes("name"),Bytes.toBytes("PIPPO"));
  - p.add(Bytes.toBytes("personal"),Bytes.toBytes("age"),Bytes.toBytes("21"));
  - hTable.put(p); hTable.close();

  - Get g = new Get(Bytes.toBytes("row1")); Result result = hTable.get(g);
  - byte [] name = result.getValue(Bytes.toBytes("personal"),Bytes.toBytes("name"));

  - Delete delete = new Delete(Bytes.toBytes("row1"));
  - delete.deleteColumn(Bytes.toBytes("personal"), Bytes.toBytes("age"));
  - hTable.delete(delete);

Hbase Mapreduce Flavor

- hbase org.apache.hadoop.hbase.mapreduce.RowCounter <tablename>

Hbase Mapreduce Program

```
Configuration config = HBaseConfiguration.create();
Job job = new Job(config, "ExampleRead");
job.setJarByClass(MyReadJob.class);     // class that contains mapper

Scan scan = new Scan();
scan.setCaching(500);        // 1 is the default in Scan, which will be bad for MapReduce jobs
scan.setCacheBlocks(false);  // don't set to true for MR jobs
// set other scan attrs
...

TableMapReduceUtil.initTableMapperJob(
  tableName,        // input HBase table name
  scan,             // Scan instance to control CF and attribute selection
  MyMapper.class,   // mapper
  null,             // mapper output key
  null,             // mapper output value
  job);
job.setOutputFormatClass(NullOutputFormat.class);   // because we aren't emitting anything from mapper

boolean b = job.waitForCompletion(true);
if (!b) {
  throw new IOException("error with job!");
}
```

Hbase Mapreduce Program Mapper

```
public static class MyMapper extends
TableMapper<Text, Text> {

  public void map(ImmutableBytesWritable row,
Result value, Context context) throws
InterruptedException, IOException {
    // process data for the row from the Result
instance.
   }
}
```

Hbase Mapreduce Program Sink and source

Count the number of distinct instances of a value in a table and write those summarized counts in another table.

```
Configuration config = HBaseConfiguration.create();
Job job = new Job(config,"ExampleSummary");
job.setJarByClass(MySummaryJob.class);     // class that contains mapper and reducer

Scan scan = new Scan();
scan.setCaching(500);        // 1 is the default in Scan, which will be bad for MapReduce jobs
scan.setCacheBlocks(false);  // don't set to true for MR jobs
// set other scan attrs

TableMapReduceUtil.initTableMapperJob(
            sourceTable,        // input table
            scan,               // Scan instance to control CF and attribute selection
            MyMapper.class,     // mapper class
            Text.class,         // mapper output key
            IntWritable.class,  // mapper output value
            job);
TableMapReduceUtil.initTableReducerJob(
            targetTable,        // output table
            MyTableReducer.class,    // reducer class
            job);
job.setNumReduceTasks(1);   // at least one, adjust as required

boolean b = job.waitForCompletion(true);
if (!b) {
            throw new IOException("error with job!");
}
```

Hbase Mapreduce Program Sink and source: Mapper

A column with a String-value is chosen as the value to summarize upon. This value is used as the key to emit from the mapper, and an IntWritable represents an instance counter.

```
public static class MyMapper extends TableMapper<Text, IntWritable>  {

        private final IntWritable ONE = new IntWritable(1);
        private Text text = new Text();

        public void map(ImmutableBytesWritable row, Result value, Context context)
throws IOException, InterruptedException {
        String val = new String(value.getValue(Bytes.toBytes("cf"),
Bytes.toBytes("attr1")));
        text.set(val);     // we can only emit Writables...

        context.write(text, ONE);
        }
}
```

Hbase Mapreduce Program Sink and source: Reducer

In the reducer, the "ones" are counted (just like any other MR example that does this), and then emits a Put.

```
public static class MyTableReducer extends TableReducer<Text, IntWritable,
ImmutableBytesWritable>  {

        public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
                int i = 0;
                for (IntWritable val : values) {
                        i += val.get();
                }
                Put put = new Put(Bytes.toBytes(key.toString()));
                put.add(Bytes.toBytes("cf"), Bytes.toBytes("count"),
Bytes.toBytes(i));

                context.write(null, put);
        }
}
```

Let’s try it !

Missing !?

- Mysql is sql complaint
  - To search data we issue commands like
    - «select columns where conditions >>
  - Generally we take advantgae of joining table
- HBASE is essentially an hashmap
  - No join of tables
  - No select

Filters

- Filters are Java classes restricting matches;
- Filter list: combines multiple filters with AND and OR
- Compare values of one or multiple columns
  - Smaller, equal, greater, substring, prefix,
- Compare metadata: column family and qualifier Qualifier prefix filter: Return (first few) matching columns
  - Column range filter: return a slice of columns (e.g. bb-bz)
- Compare names of rows Note: it is preferable to use scan options

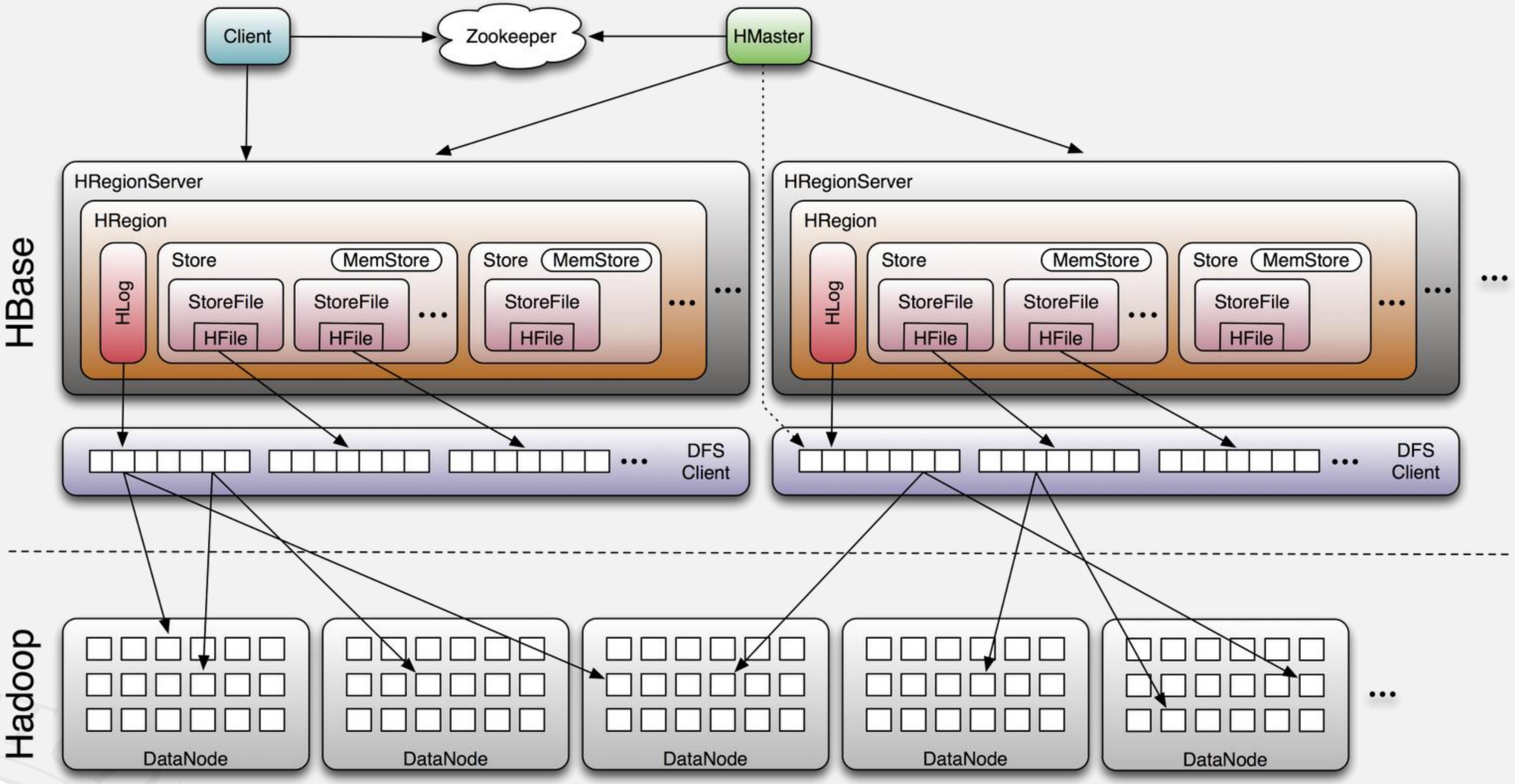
Scan advanced

- scan ’student’,{ FILTER => "KeyOnlyFilter()"}
- scan ’student’,{ FILTER => "(PrefixFilter (’pi’)) AND MultipleColumnPrefixFilter(’a’,’b’) AND ColumnCountGetFilter(2)" }
- scan ’student’,{ COLUMNS=>[’a:age’], FILTER => "SingleColumnValueFilter(’a’,’age’,>,’binary:19’)"}

HBase Components

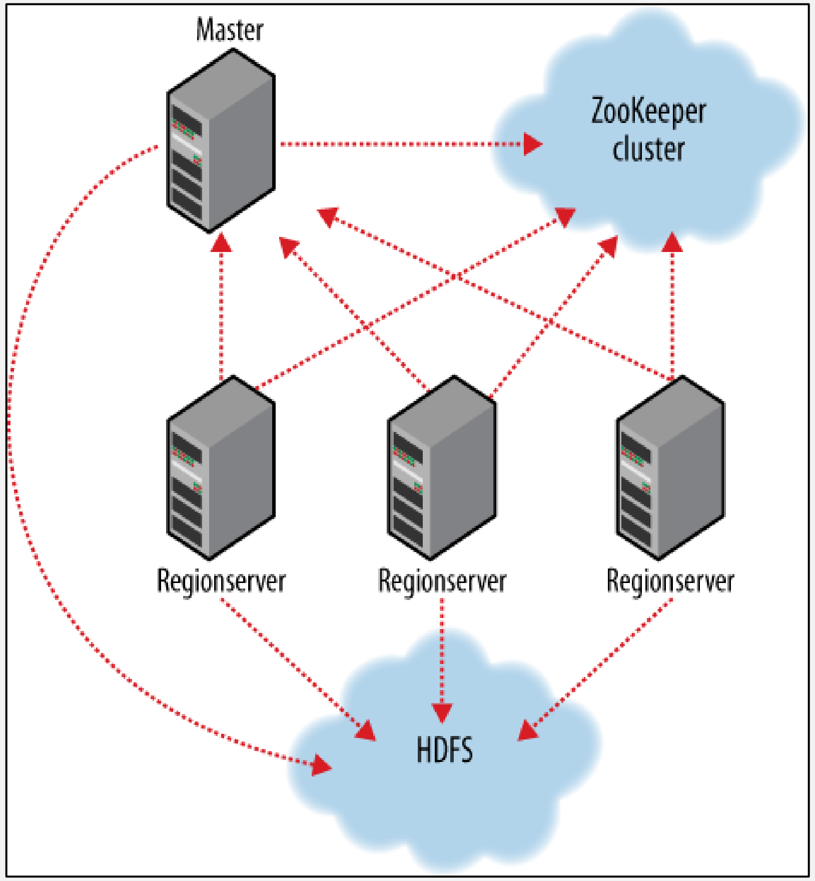
- **Region**
  - A subset of a table's rows, like horizontal range partitioning
  - Automatically done
- **RegionServer (many slaves)**
  - Manages data regions
  - Serves data for reads and writes (***using a log***)
- **Master**
  - Responsible for coordinating the slaves
  - Assigns regions, detects failures
  - Admin functions

Big Picture


Client
Zookeeper
HMaster
HBase
HRegionServer
HRegion
HLog
Store
MemStore
StoreFile
HFile
...
DFS Client
Hadoop
DataNode

ZooKeeper

- HBase depends on ZooKeeper
- By default HBase manages the ZooKeeper instance
  - E.g., starts and stops ZooKeeper
- HMaster and HRegionServers register themselves with ZooKeeper

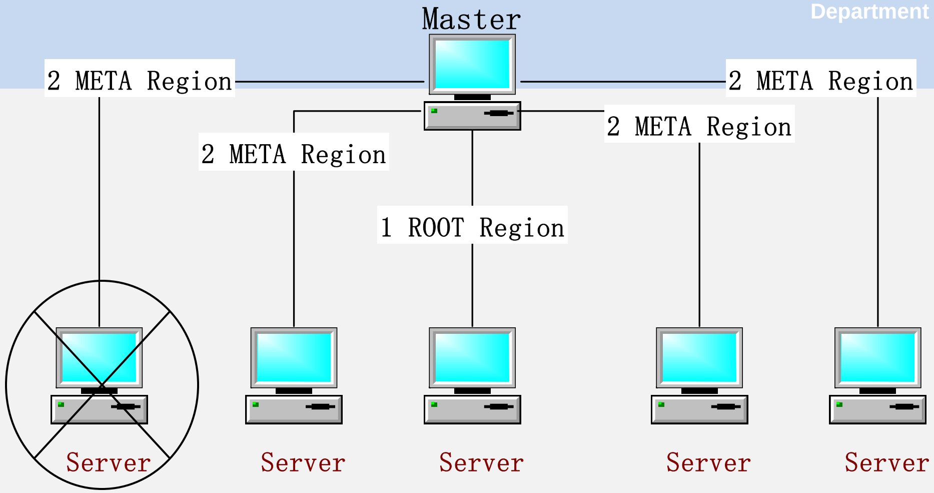


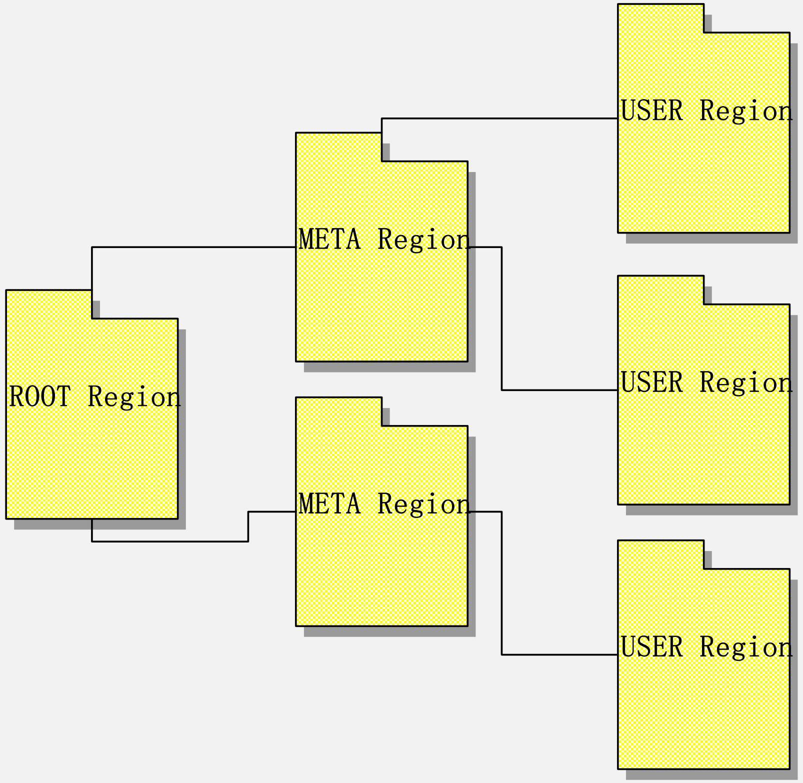
HBaseMaster

- Assign regions to HRegionServers.

1. ROOT region locates all the META regions.
2. META region maps a number of user regions.
3. Assign user regions to the HRegionServers.

- Enable/Disable table and change table schema
- Monitor the health of each Server





ROOT/META Table

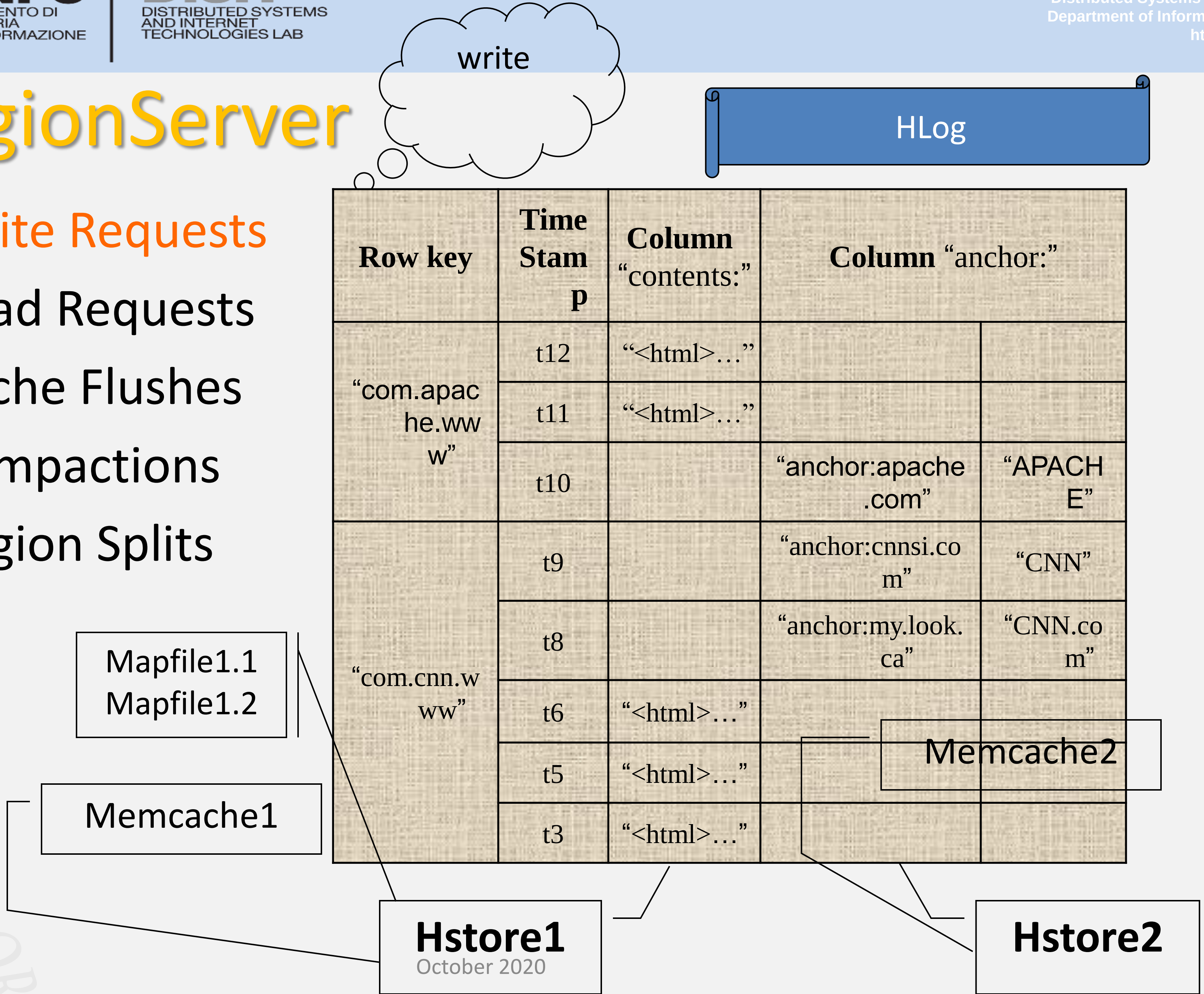
- Each row in the ROOT and META tables is approximately 1KB in size. At the default size of 256MB.

$$
\begin{aligned}
&1\ ROOT\ table\\
&= 2^{18}\ META\ regions\\
&= 2^{18} \times 2^{18} USER\ regions\\
&= 2^{54} KB = 2^{64} bytes
\end{aligned}
$$

2^{24}TB

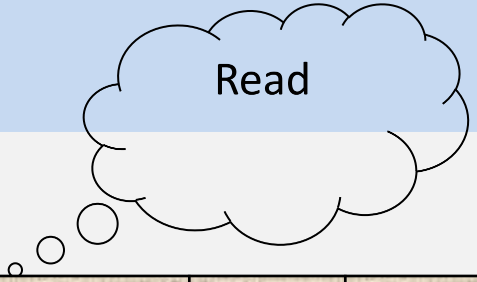
HRegionServer

- Write Requests
- Read Requests
- Cache Flushes
- Compactions
- Region Splits



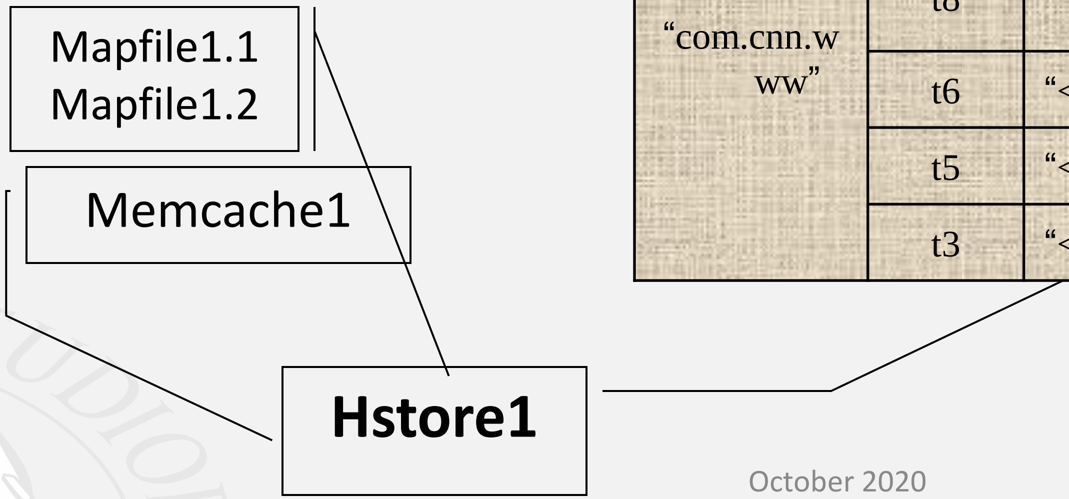
Row key	Time Stamp	Column “contents:”	Column “anchor:”	
“com.apache.www”	t12	“<html>…”		
	t11	“<html>…”		
	t10		“anchor:apache.com”	“APACHE”
“com.cnn.www”	t9		“anchor:cnnsi.com”	“CNN”
	t8		“anchor:my.look.ca”	“CNN.com”
	t6	“<html>…”		
	t5	“<html>…”		
	t3	“<html>…”		

HRegionServer



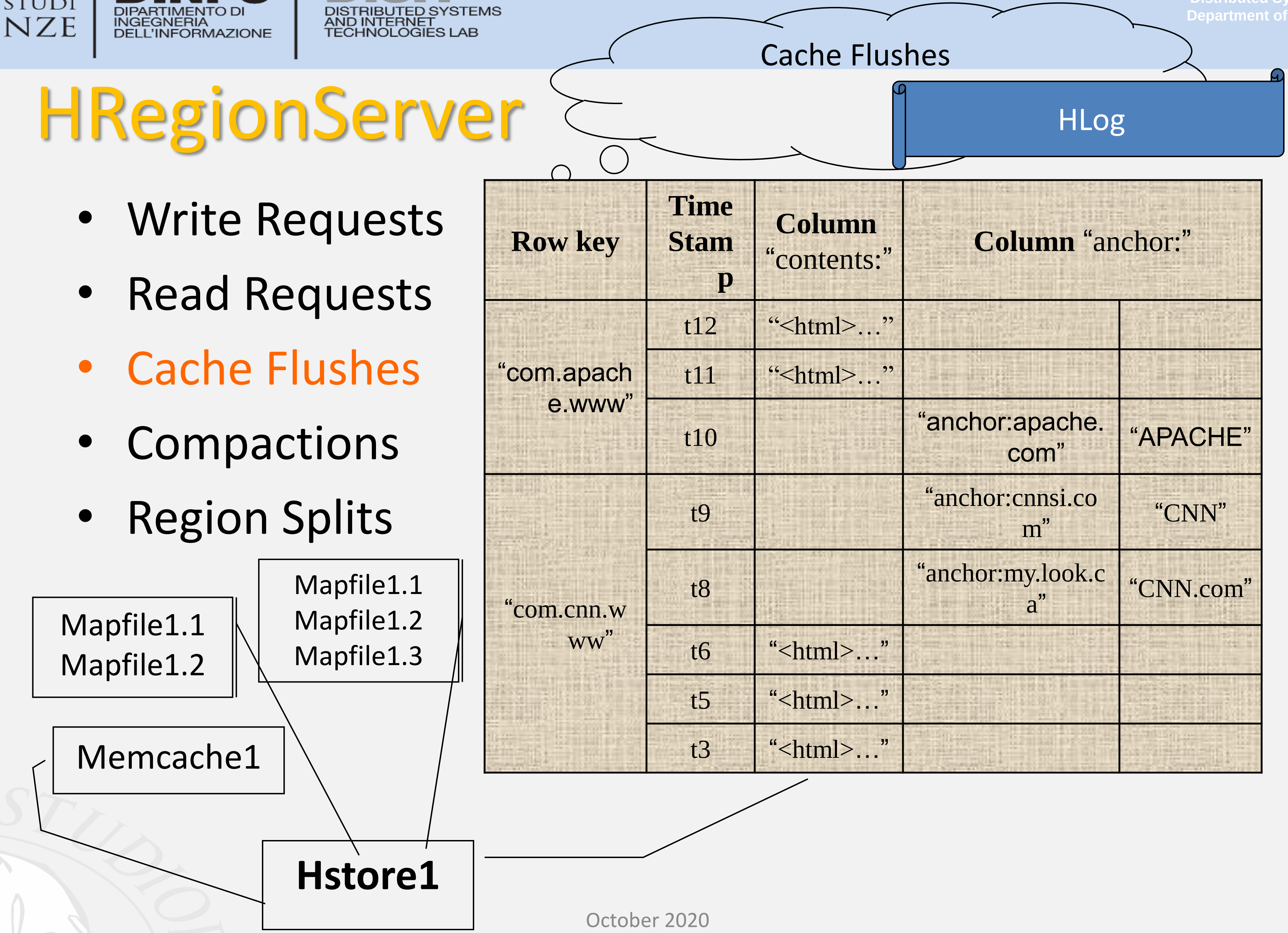
- Write Requests
- Read Requests
- Cache Flushes
- Compactions
- Region Splits

Row key	Time Stamp	Column "contents:"	Column "anchor:"	
"com.apache.www"	t12	"<html>…"		
	t11	"<html>…"		
	t10		"anchor:apache.com"	"APACHE"
"com.cnn.www"	t9		"anchor:cnnsi.com"	"CNN"
	t8		"anchor:my.look.ca"	"CNN.com"
	t6	"<html>…"		
	t5	"<html>…"		
	t3	"<html>…"		



HRegionServer

- Write Requests
- Read Requests
- Cache Flushes
- Compactions
- Region Splits



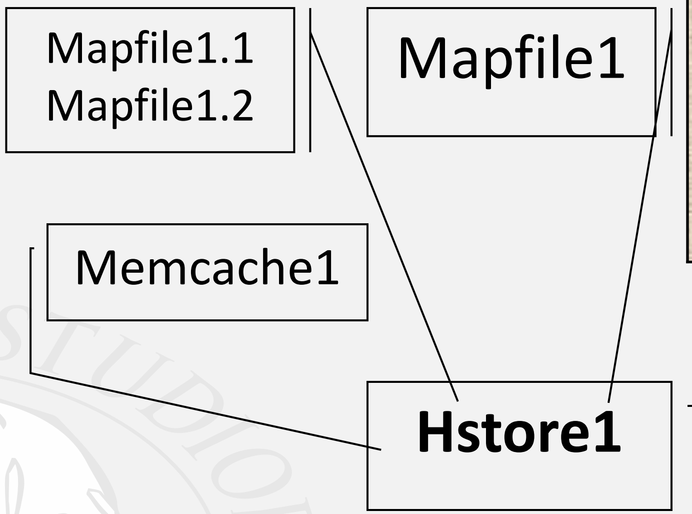
Row key	Time Stamp	Column "contents:"	Column "anchor:"	
"com.apache.www"	t12	"<html>…"		
	t11	"<html>…"		
	t10		"anchor:apache.com"	"APACHE"
"com.cnn.www"	t9		"anchor:cnnsi.com"	"CNN"
	t8		"anchor:my.look.ca"	"CNN.com"
	t6	"<html>…"		
	t5	"<html>…"		
	t3	"<html>…"		

HRegionServer

- Write Requests
- Read Requests
- Cache Flushes
- Compactions
- Region Splits

Compactions

Row key	**Time Stamp**	**Column** "contents:"	**Column** "anchor:"	
"com.apache.www"	t12	"<html>…"		
	t11	"<html>…"		
	t10		"anchor:apache.com"	"APACHE"
"com.cnn.www"	t9		"anchor:cnnsi.com"	"CNN"
	t8		"anchor:my.look.ca"	"CNN.com"
	t6	"<html>…"		
	t5	"<html>…"		
	t3	"<html>…"		

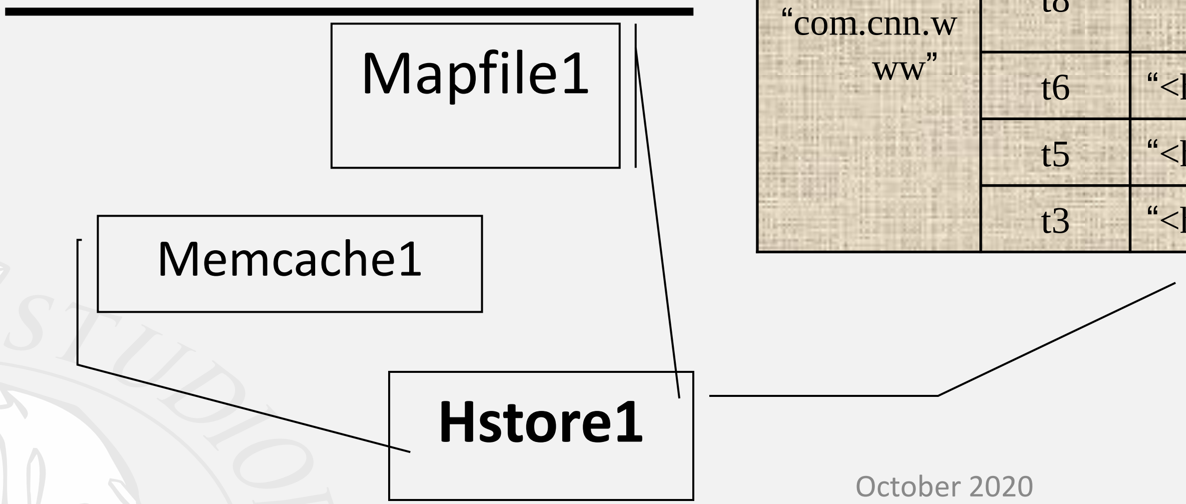


HRegionServer

- Write Requests
- Read Requests
- Cache Flushes
- Compactions
- Region Splits

Region Splits

Row key	Time Stamp	Column "contents:"	Column "anchor:"	
"com.apache.www"	t12	"<html>…"		
	t11	"<html>…"		
	t10		"anchor:apache.com"	"APACHE"
"com.cnn.www"	t9		"anchor:cnnsi.com"	"CNN"
	t8		"anchor:my.look.ca"	"CNN.com"
	t6	"<html>…"		
	t5	"<html>…"		
	t3	"<html>…"		



Agenda

- Monitoring
- Apache HBASE
- Case studio
- Apache Phoenix
- Apache Solr
- Elastic Search

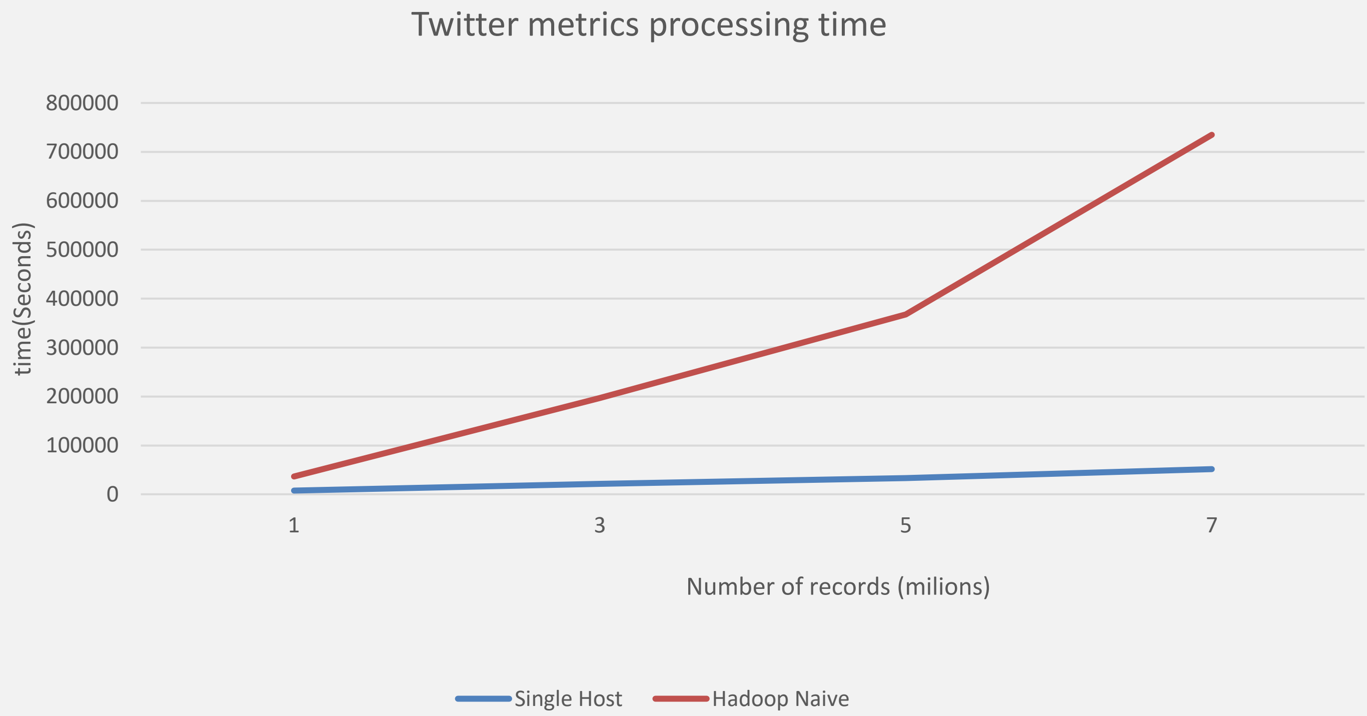
Solution

- Migrating to a distribuited architcure
  - We have to rethink the problem in parallel way

Sequential minded approach

- For each search execute a MR job
  - If a MR job takes 1 minute (ideally) time and i have to compute for 1000 -> i have to wait 1000 minute !

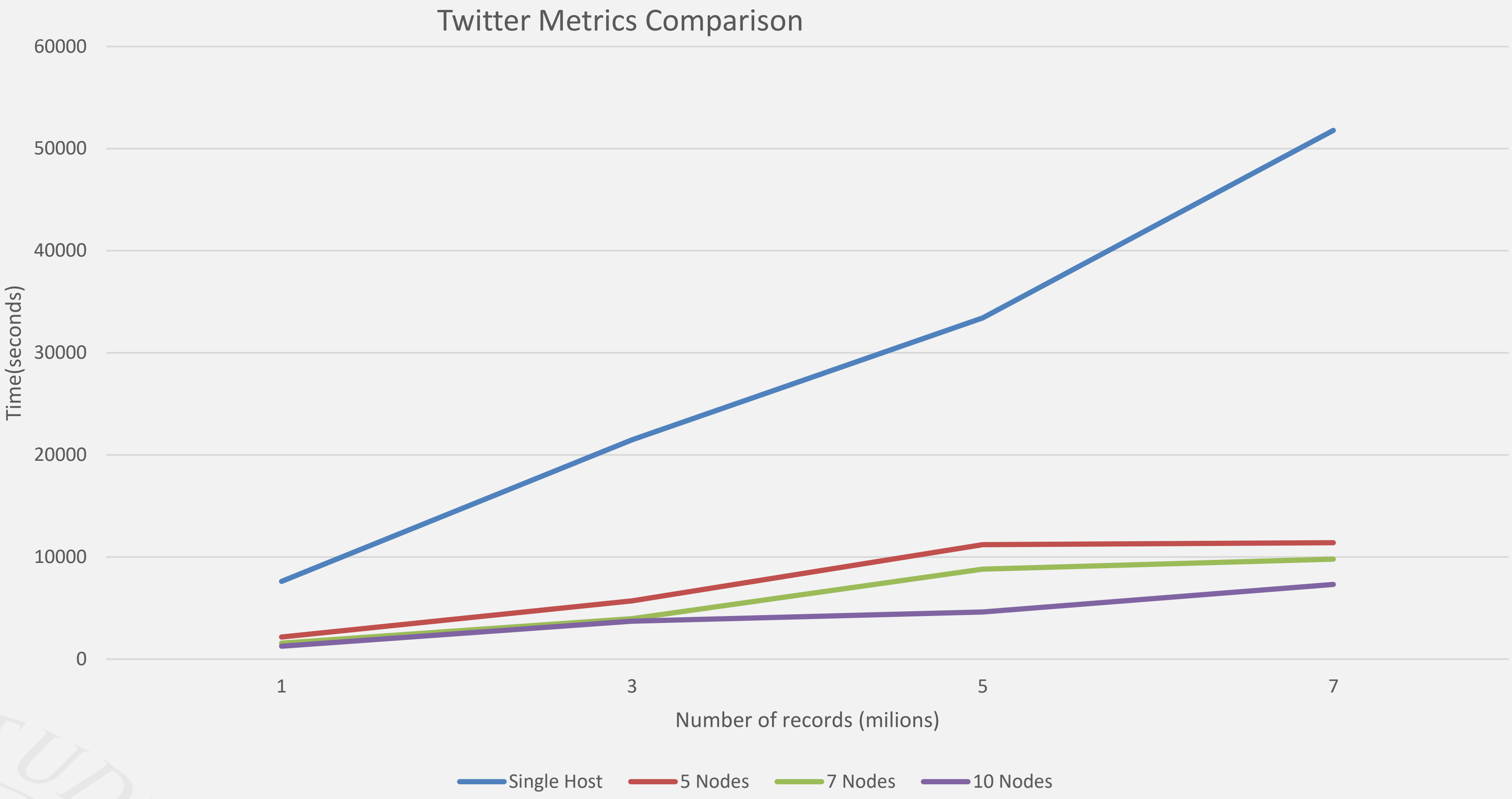
Single Host vs Hadoop



Parallel-computing minded aproach

- Scan all records and counts each word sorting per day
- As final step consider only the words involded in our search

Single Host vs Hadoop



All Quiet on the Western Front …

Google calls it:	Hadoop equivalent:
MapReduce	Hadoop
GFS	HDFS
Bigtable	HBase
Chubby	Zookeeper

- 1979: First commercial SQL RDBMs
- 1990: Transaction processing on SQL now popular
- 2006: Hadoop and other “big data” technologies
- 2008: NoSQL
- 2011: SQL on Hadoop
- 2014: Interactive analytics on Hadoop and NoSQL with SQL (Phoenix)

Some References

I. Nesi, Paolo, Gianni Pantaleo, and Gianmarco Sanesi. "A hadoop based platform for natural language processing of web pages and documents." *Journal of Visual Languages & Computing* 31 (2015): 130-138.

II. Michael, Maged, et al. "Scale-up x scale-out: A case study using nutch/lucene." *2007 IEEE International Parallel and Distributed Processing Symposium*. IEEE, 2007.

III. Appuswamy, Raja, et al. "Scale-up vs scale-out for hadoop: Time to rethink? " *Proceedings of the 4th annual Symposium on Cloud Computing*. ACM, 2013.

- Apache Hadoop ver. 3.0.0
- Apache Hbase ver 1.2.6
- Apache Phoenix ver 4.14.0

Agenda

- Monitoring
- Apache HBASE
- Case studio
- Apache Phoenix
- Apache Solr
- Elastic Search

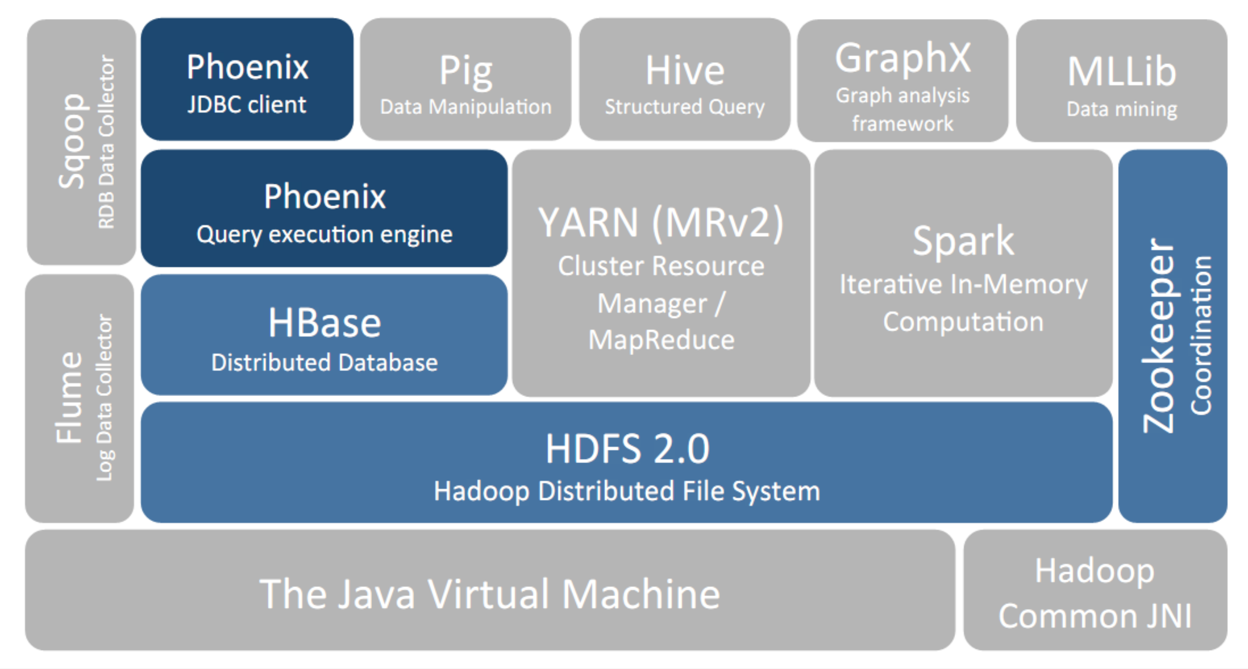
Why

- Hbase is Ok BUT
  - It takes too much expertise to write an application
  - It takes too much code to do anything
  - Your application is tied too closely with your data model

Apache Phoenix

- **What**: A relational database layer for Apache Hbase
  - Low Latency Query Model & SQL support over HBase API
  - SQL query compiled it into a series of HBase scans
  - Metadata is stored in an HBase table and versioned
  - Pushes Computation to the HBase Region Servers
- **How: Coprocessors** (Server-Side) **Minimize Data Transfer** Uses **Native HBase APIs** Not Map/Reduce framework of Hbase
- A JDBC driver

Apache Phoenix



Performance

- Pushes down computatioon to region servers
  - Start/stop key range(s)
  - Time range min/max
  - Predicates
  - Aggregation
  - Sort
  - Limit
- Parallelizes query from client
  - Intra-region through staBsBcs collecBon
- Supports secondary indexes
  - Global & co-located

Agenda

- Monitoring
- Apache HBASE
- Case studio
- Apache Phoenix
- Apache Solr
- Elastic Search

Overview

- Introduction to Lucene & Solr
  - Getting started
  - Indexing using Solr
  - Updating & deleting files
  - Searching using Solr

What is Lucene?

- Lucene is An open source Java-based IR library enabling text based search

What is Solr

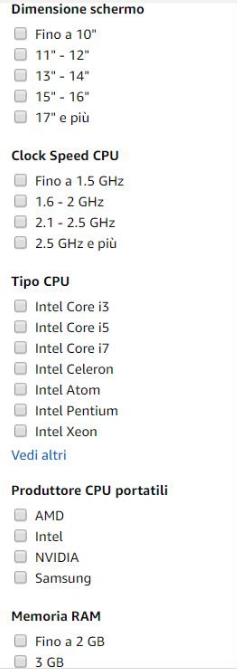
- Solr is:
  - An open source enterprise search server
  - Based on the Lucene Java search library
  - A web based application that processes HTTP request and returns HTTP responses
  - completed with XML/HTTP APIs, caching, replication, and web administration interface.

Why solr

- Using many Lucene best practices
- Easy setup, configuration and Easy to extend
- Providing faceted navigation, spell checking, highlighting, clustering, grouping, & any other search features
- Supporting clients in:
  - HTTP
  - Java
  - Python
  - PHP
  - Ruby
  - JSON

Why Solr

- Some reasons of using Solr:
  - Good indexing performance
  - SolrCloud feature (Solr 4.x above)
  - Multi-Faceted searches
  - Geospatial searches
  - Who uses Lucene/Solr?
    - Cisco, ebay, Boeing, AT& T, Ford and many, many others...!



Dimensione schermo
- □ Fino a 10"
- □ 11" - 12"
- □ 13" - 14"
- □ 15" - 16"
- □ 17" e più

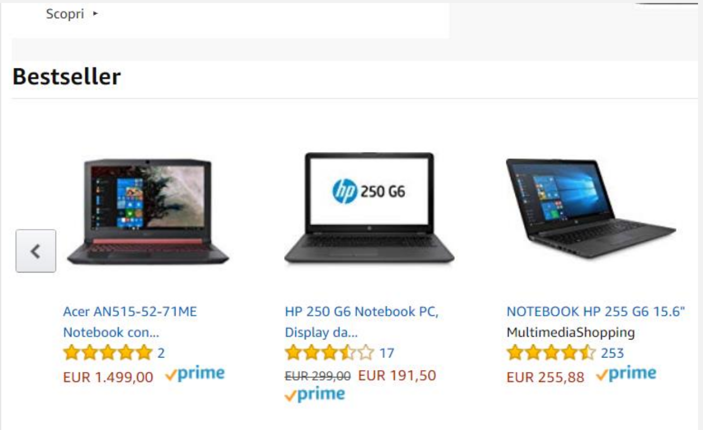
Clock Speed CPU
- □ Fino a 1.5 GHz
- □ 1.6 - 2 GHz
- □ 2.1 - 2.5 GHz
- □ 2.5 GHz e più

Tipo CPU
- □ Intel Core i3
- □ Intel Core i5
- □ Intel Core i7
- □ Intel Celeron
- □ Intel Atom
- □ Intel Pentium
- □ Intel Xeon

Vedi altri

Produttore CPU portatili
- □ AMD
- □ Intel
- □ NVIDIA
- □ Samsung

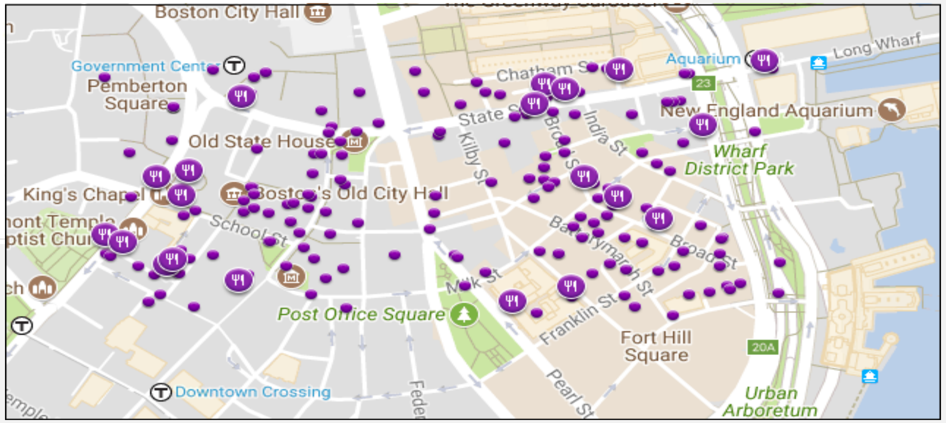
Memoria RAM
- □ Fino a 2 GB
- □ 3 GB



Scopri ▸

Bestseller

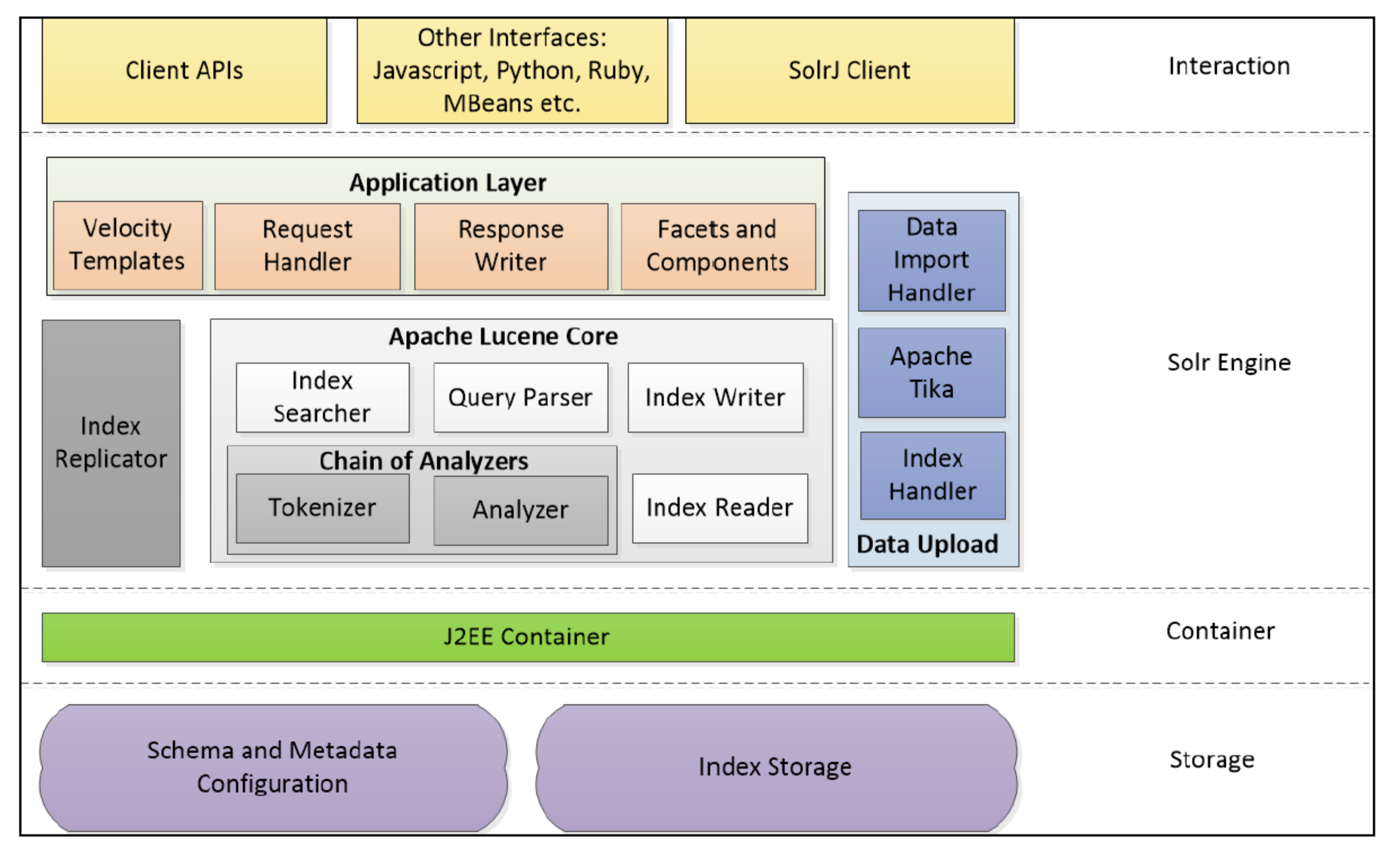
Acer AN515-52-71ME Notebook con...	HP 250 G6 Notebook PC, Display da...	NOTEBOOK HP 255 G6 15.6" MultimediaShopping
★★★★★ 2	★★★½☆ 17	★★★★½ 253
EUR 1.499,00 ✓prime	~~EUR 299,00~~ EUR 191,50 ✓prime	EUR 255,88 ✓prime



Comparison to Database Technology

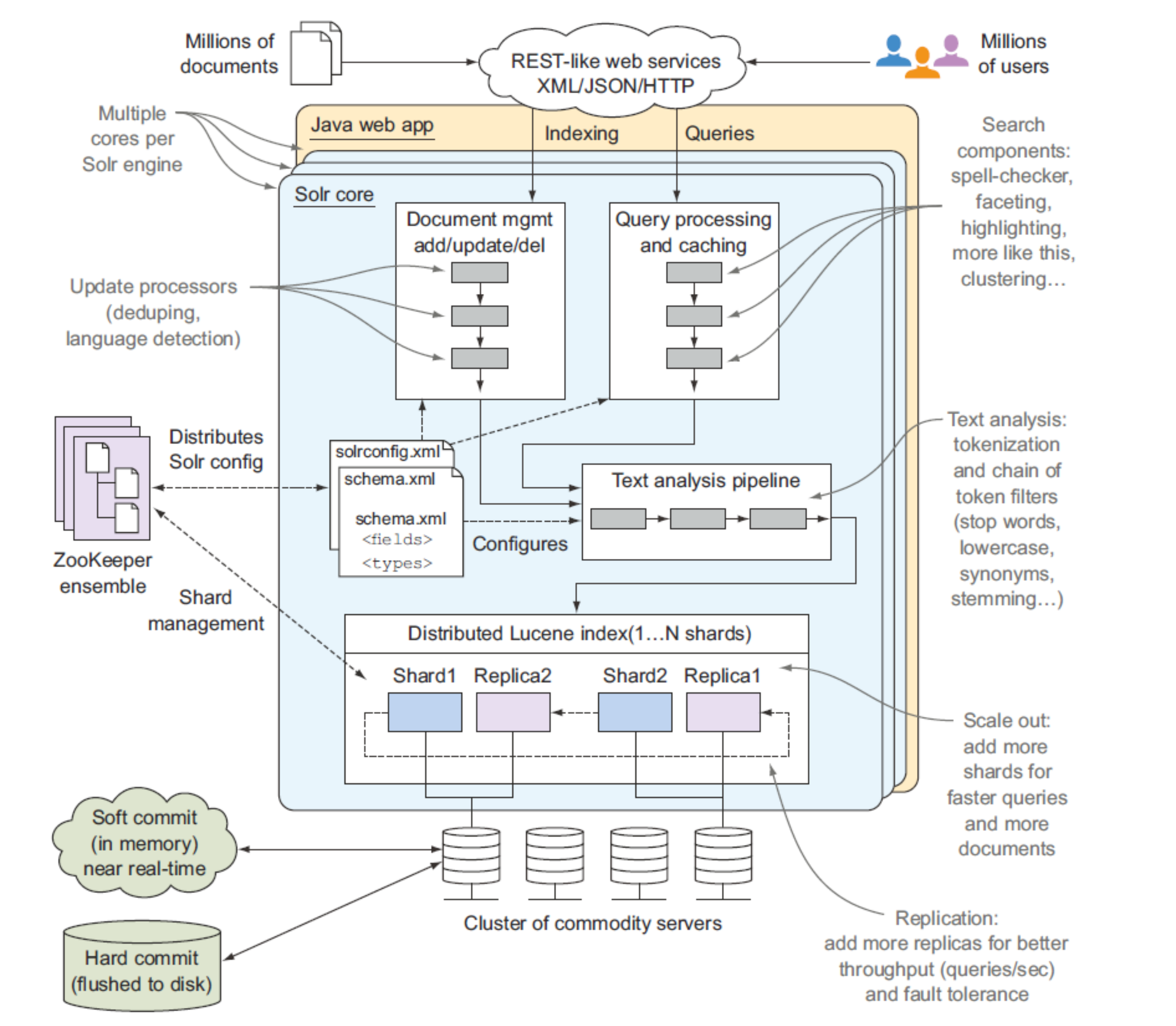
- The most important comparison to make is the data model
  - Data model is the organizational structure of data
- RDBMS:
  - Its data model is based on multiple tables with lookup keys between them
  - A join capability for querying across tables
  - A flexible data model
- Lucene Solr:
  - Has a document oriented data model
    - Analogous to a single table without join possibilities
    - Document-oriented databases have a rich nested structure similar to XML/JSON → MongoDB (NoSQL)
- Has a flat document structure
  - Supporting multi-valued fields with an array of values

Solr Architecture



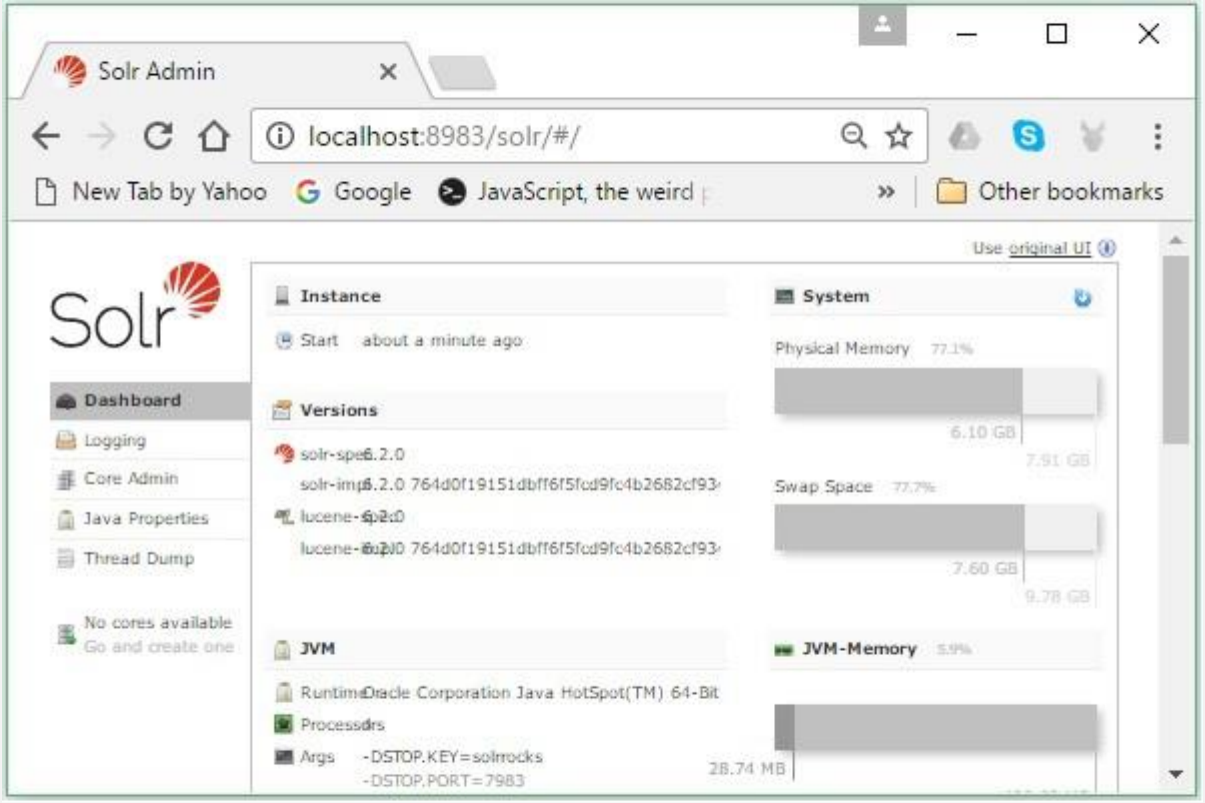
Source: «Scaling Big Data with Hadoop and Solr», (H. V. Karambelkar) - Packt Publishing Ed. (2015)

Solr Architecture


Millions of documents
REST-like web services XML/JSON/HTTP
Millions of users
Multiple cores per Solr engine
Java web app
Indexing
Queries
Search components: spell-checker, faceting, highlighting, more like this, clustering…
Solr core
Document mgmt add/update/del
Query processing and caching
Update processors (deduping, language detection)
Distributes Solr config
ZooKeeper ensemble
solrconfig.xml
schema.xml
schema.xml
<fields>
<types>
Configures
Text analysis pipeline
Text analysis: tokenization and chain of token filters (stop words, lowercase, synonyms, stemming…)
Shard management
Distributed Lucene index(1…N shards)
Shard1
Replica2
Shard2
Replica1
Scale out: add more shards for faster queries and more documents
Soft commit (in memory) near real-time
Hard commit (flushed to disk)
Cluster of commodity servers
Replication: add more replicas for better throughput (queries/sec) and fault tolerance

Source: «Solr in Action», (T. Greinger, T. Potter) – Manning Ed. (2014)

Solr Interface & Functionalities



Start about a minute ago

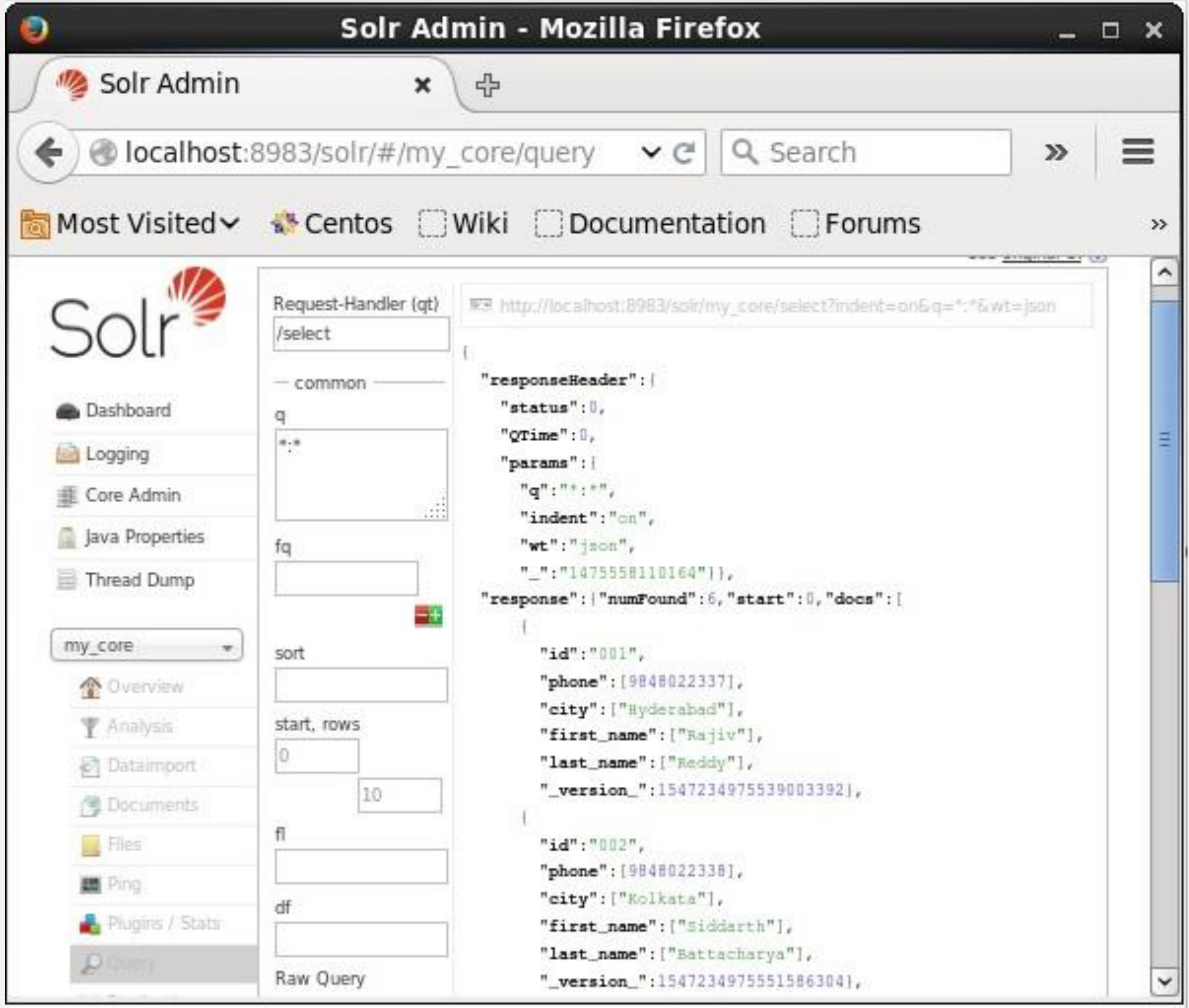
Physical Memory 77.1% 6.10 GB 7.91 GB

solr-spec 6.2.0
solr-impl 6.2.0 764d0f19151dbff6f5fcd9fc4b2682cf93
lucene-spec 6.2.0
lucene-impl 6.2.0 764d0f19151dbff6f5fcd9fc4b2682cf93

Swap Space 77.7% 7.60 GB 9.78 GB

Runtime Oracle Corporation Java HotSpot(TM) 64-Bit
Processors
Args -DSTOP.KEY=solrrocks
-DSTOP.PORT=7983

JVM-Memory 5.9% 28.74 MB



Request-Handler (qt): /select

common

q: *:*

fq:

sort:

start, rows: 0 10

fl:

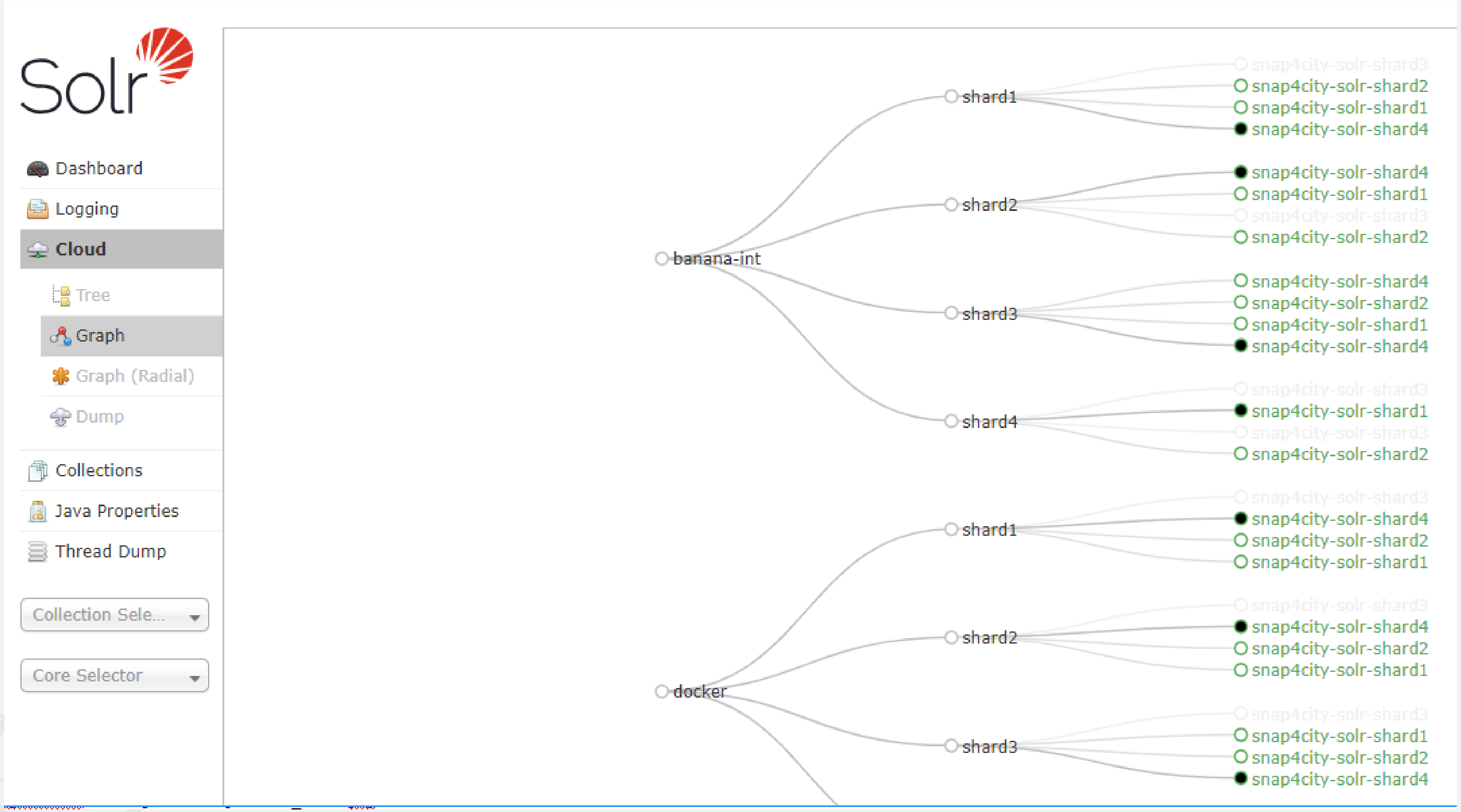
df:

Raw Query

http://localhost:8983/solr/my_core/select?indent=on&q=*:*&wt=json

```
{
  "responseHeader":{
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"on",
      "wt":"json",
      "_":"1475558110164"}},
  "response":{"numFound":6,"start":0,"docs":[
      {
        "id":"001",
        "phone":[9848022337],
        "city":["Hyderabad"],
        "first_name":["Rajiv"],
        "last_name":["Reddy"],
        "_version_":1547234975539003392},
      {
        "id":"002",
        "phone":[9848022338],
        "city":["Kolkata"],
        "first_name":["Siddarth"],
        "last_name":["Battacharya"],
        "_version_":1547234975551586304},
```

Solr Interface & Functionalities



Solr API

COLLECTIONS API

```
create -c NewC
curl http://localhost:8983/solr/admin/cores?action=CREATE&name=NewC&instanceDir=NewC
```

INSERT / UPDATE API

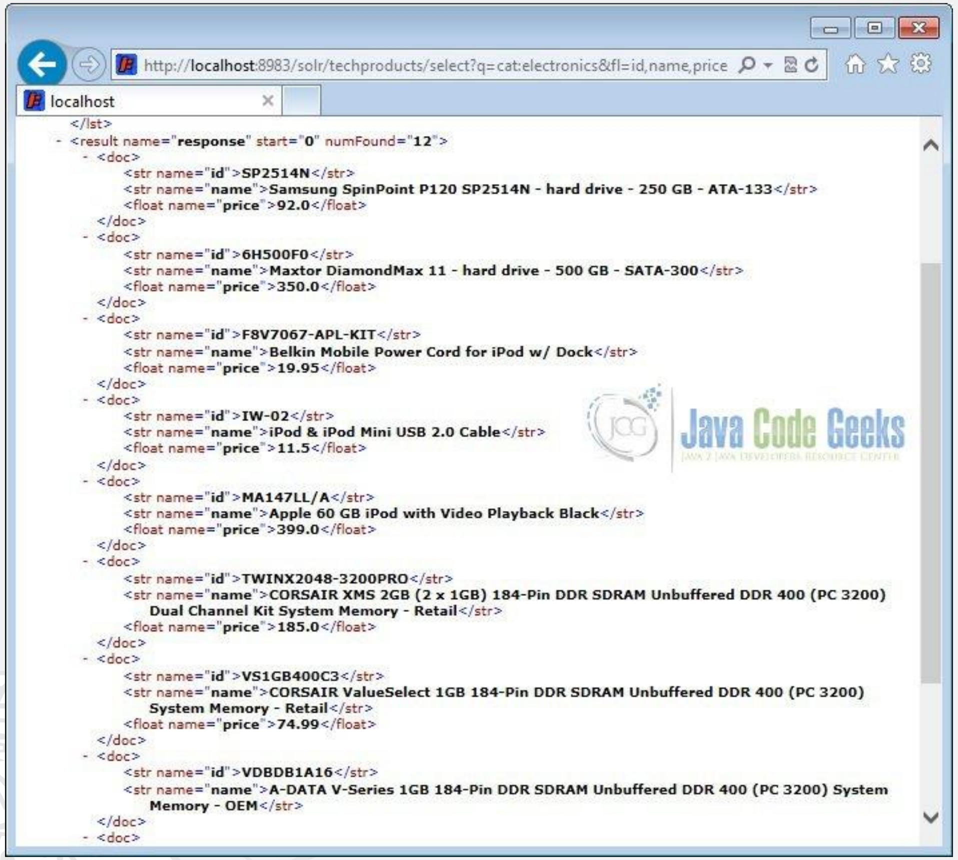
```
curl http://localhost:8983/solr/NewC/update -d "[
{            \"id\" : \"book1\",
             \"title_t\" : \"Solr In Action\",
             \"author_s\" : \"Timothy Potter\"
}]"
```

SEARCH API

```
http://localhost:8983/solr/<COLLECTION>/select?q=<QUERY>
curl http://localhost:8983/solr/techproducts/select?q=<KEYWORD>
curl http://localhost:8983/solr/techproducts/select?q=cat:electronics
```

Solr API

http://localhost:8983/solr/techproducts/select?q=cat:electronics&fl=id,name,price


http://localhost:8983/solr/techproducts/select?q=cat:electronics&fl=id,name,price
localhost
Java Code Geeks
JAVA 2 JAVA DEVELOPERS RESOURCE CENTER

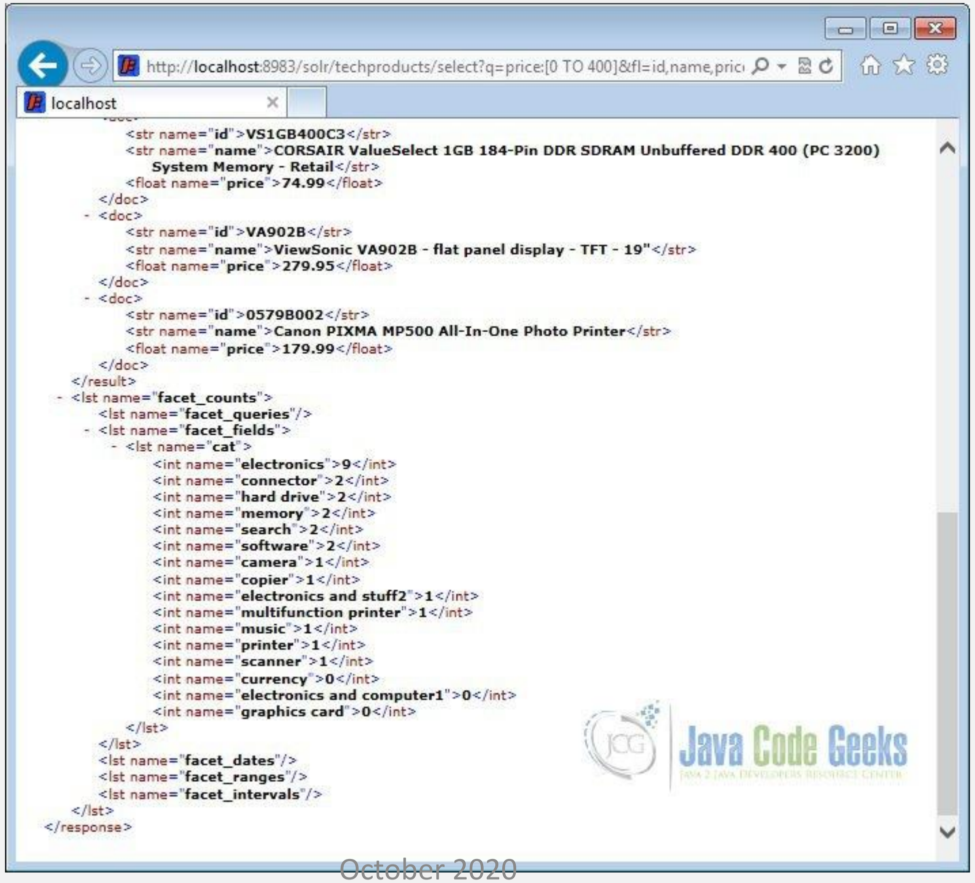
```
    </lst>
  - <result name="response" start="0" numFound="12">
    - <doc>
        <str name="id">SP2514N</str>
        <str name="name">Samsung SpinPoint P120 SP2514N - hard drive - 250 GB - ATA-133</str>
        <float name="price">92.0</float>
      </doc>
    - <doc>
        <str name="id">6H500F0</str>
        <str name="name">Maxtor DiamondMax 11 - hard drive - 500 GB - SATA-300</str>
        <float name="price">350.0</float>
      </doc>
    - <doc>
        <str name="id">F8V7067-APL-KIT</str>
        <str name="name">Belkin Mobile Power Cord for iPod w/ Dock</str>
        <float name="price">19.95</float>
      </doc>
    - <doc>
        <str name="id">IW-02</str>
        <str name="name">iPod & iPod Mini USB 2.0 Cable</str>
        <float name="price">11.5</float>
      </doc>
    - <doc>
        <str name="id">MA147LL/A</str>
        <str name="name">Apple 60 GB iPod with Video Playback Black</str>
        <float name="price">399.0</float>
      </doc>
    - <doc>
        <str name="id">TWINX2048-3200PRO</str>
        <str name="name">CORSAIR XMS 2GB (2 x 1GB) 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200)
          Dual Channel Kit System Memory - Retail</str>
        <float name="price">185.0</float>
      </doc>
    - <doc>
        <str name="id">VS1GB400C3</str>
        <str name="name">CORSAIR ValueSelect 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200)
          System Memory - Retail</str>
        <float name="price">74.99</float>
      </doc>
    - <doc>
        <str name="id">VDBDB1A16</str>
        <str name="name">A-DATA V-Series 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System
          Memory - OEM</str>
      </doc>
    - <doc>
```

Solr Quert Parameters

- **qt** – Query handler for the request. Standard query handler is used if not specified.
- **q** – It is used to specify the query event.
- **fq** – Used to specify filter queries.
- **sort** – Used to sort the results in ascending or descending order.
- **start, rows** – start specifies the staring number of the result set. By default it is zero. rows specify the number of records to return.
- **fl** – Used to return selective fields.
- **wt** – Specifies the response format. Default is XML.
- **indent** – Setting to true makes the response more readable.
- **debugQuery** – Setting the parameter to true gives the debugging information as part of response.
- **dismax** – To specify the dismax parser.
- **edismax** – To specify the edismax parser.
- **facet** – Setting to true enables the faceting.
- **spatial** – Used for geospatial searches.
- **spellcheck** – Setting to true help in searching similar terms.

Solr Faceted Search

http://localhost:8983/solr/techproducts/select?q=price:[0 TO 400]&fl=id,name,price&facet=true&facet.field=cat


http://localhost:8983/solr/techproducts/select?q=price:[0 TO 400]&fl=id,name,pric
localhost
Java Code Geeks
JAVA 2 JAVA DEVELOPERS RESOURCE CENTER

```
      <str name="id">VS1GB400C3</str>
      <str name="name">CORSAIR ValueSelect 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200)
        System Memory - Retail</str>
      <float name="price">74.99</float>
    </doc>
  - <doc>
      <str name="id">VA902B</str>
      <str name="name">ViewSonic VA902B - flat panel display - TFT - 19"</str>
      <float name="price">279.95</float>
    </doc>
  - <doc>
      <str name="id">0579B002</str>
      <str name="name">Canon PIXMA MP500 All-In-One Photo Printer</str>
      <float name="price">179.99</float>
    </doc>
  </result>
- <lst name="facet_counts">
    <lst name="facet_queries"/>
  - <lst name="facet_fields">
    - <lst name="cat">
        <int name="electronics">9</int>
        <int name="connector">2</int>
        <int name="hard drive">2</int>
        <int name="memory">2</int>
        <int name="search">2</int>
        <int name="software">2</int>
        <int name="camera">1</int>
        <int name="copier">1</int>
        <int name="electronics and stuff2">1</int>
        <int name="multifunction printer">1</int>
        <int name="music">1</int>
        <int name="printer">1</int>
        <int name="scanner">1</int>
        <int name="currency">0</int>
        <int name="electronics and computer1">0</int>
        <int name="graphics card">0</int>
      </lst>
    </lst>
    <lst name="facet_dates"/>
    <lst name="facet_ranges"/>
    <lst name="facet_intervals"/>
  </lst>
</response>
```

Agenda

- Monitoring
- Apache HBASE
- Apache Phoenix
- Case studio
- Apache Solr
- Elastic Search

Elasticsearch

- real time
- Search & Analytics Engine
- Distributed
- Scales massively
- High availability
- Restful api
- Json over HTTP
- Schema free
- Multi tenancy
- Open source
- Lucene based

API

- curl –X GET localhost:9200/?pretty
  - Input Data
  - Retrieve Data
  - Update Data
  - Delete Data

Input Data

PUT /index/type/id

```
PUT /myapp/tweet/1 -d '
{ "tweet": “A tweet text”,
"nick": "@twitternick",
"name": “Pippo",
'
```

Retrieve Data

- GET /myapp/tweet/1

```
{
 "_index": "myapp",
 "_type": "tweet",
 "_id": "1",
 "_version": 1,
 "exists": true,
 "_source": { ...OUR
TWEET... }
}
```

Update Data

- PUT /myapp/tweet/1 -d '

```
{
"tweet": "I know #elasticsearch is AWESOME",
"nick": "@clintongormley",
"name": "Clinton Gormley",
"date": "2013-06-03",
"rt": 5,
"loc": {
 "lat": 13.4,
 "lon": 52.5
}
}
'
```

```
{
 "_index": "myapp",
 "_type": "tweet",
 "_id": "1",
 "_version": 2,
 "ok": true
}

# atomic delete and put
```

Delete Data

- DELETE /myapp/tweet/1

```
{
 "_index": "myapp",
 "_type": "tweet",
 "_id": "1",
 "_version": 3,
 "ok": true,
 "found": true
}
```

RDBMS comparison

SQL Server manage Databases i.e. Tables i.e. Columns/Rows

RDBMS comparison

Elastic Search manage Indices i.e. Types i.e. Documents with Properties

An Elastic Search cluster can contain multiple **Indices** (databases), which in turn contain multiple **Types** (tables). These types hold multiple **Documents** (rows), and each document has **Properties**(columns).

Glossary

- Node: A node is a running instance of elasticsearch which belongs to a *cluster.*
- Shard: A shard is a single Lucene instance. It is a low-level “worker” unit which is managed automatically by elasticsearch. An index is a logical namespace which points to primary and replica shards.
- Primary Shard: Each document is stored in a single *primary shard*. When you index a document, it is *indexed* first on the *primary shard*, then on all *replicas* of the primary shard.
- Replica Shard: Each *primary shard* can have zero or more *replicas*. A replica is a copy of the primary shard, and has two purposes: a) increase fail over b) increase performance

Glossary

- Index: An index is like a *database* in a relational database.
- Type: A type is like a *table* in a relational database. Each type has a list of fields that can be specified for documents of that type.
- Document: JSON document which is stored in elasticsearch. It is like a row in a table in a relational database.
- Field: A *document* contains a list of fields, or key-value pairs. The value can be a simple (scalar) value (eg a string, integer, date), or a nested structure like an array or an object. A field is similar to a column in a table in a relational database.
- Mapping: mapping is like a *schema definition* in a relational database.
  The mapping defines how each field in the document is *analyzed*.
- Routing: When you index a document, it is stored on a single primary shard. That shard is chosen by hashing the *routing* value. By default the *routing* value is derived from the ID of the document.

Core Field Types

- Strings: string
- Datetimes: date
- Whole numbers: byte, short, integer, long
- Floats: float, double
- Booleans: boolean
- Objects: object

- Also: multi_field, ip, geo_point, geo_shape,

Auto Detection of Field

"foo bar"	string
"2018-10-09"	date
10	byte, short, integer, long
10.0	float, double
true	boolean
{ foo: "bar" }	object
["foo","bar"]	No special mapping. Any field can have multi-values

Some more Glossary

- Term: A term is an exact value that is indexed in elasticsearch. The terms foo, Foo, FOO are NOT equivalent.
- Text: Text (or full text) is ordinary unstructured text, such as this paragraph. By default, text will be analyzed into terms, which is what is actually stored in the index. Text fields need to be analyzed at index time in order to be searchable as full text, and keywords in full text queries must be analyzed at search time to produce (and search for) the same terms that were generated at index time.
- Analysis: Analysis is the process of converting *full text* to *terms*. Depending on which analyzer is used, these phrases: FOO BAR, Foo-Bar, foo,bar will probably all result in the terms foo and bar. These terms are what is actually stored in the index.

Some more glossary (Cont.)

- Tokenizer: Tokenizers are used to break a string down into a stream of terms or tokens. A simple tokenizer might split the string up into terms wherever it encounters whitespace or punctuation.
- Facets: They enable you to calculate and summarize data about the current query on-the-fly. They can be used for all sorts of tasks such as dynamic counting of result values or even distribution histograms. Facets only perform their calculations one-level deep, and they cant be easily combined.
- Aggregations: Aggregations are similar to facets in many ways, and overcome the limitations of facets. Indeed, aggregations are meant to eventually replace facets altogether. Facets are and should be considered deprecated and will likely be removed in one of the future major releases. One of the major limitations of facets is that you can't have facets of facets. Which is to say, facets cannot be nested. The ability to nest aggregations therefore brings a great deal of power that was missing in facets.
  - The two broad families of aggregations are metrics aggregations and bucket aggregations. Metrics aggregations calculate some value (like an average) over a set of documents, and bucket aggregations group documents into buckets.

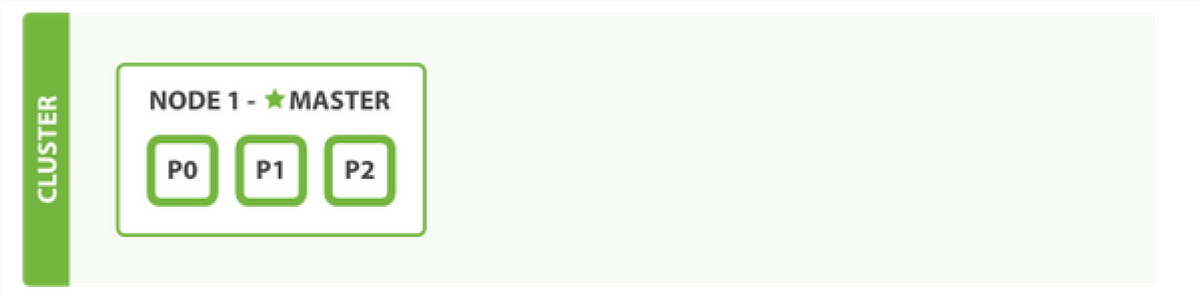
- Schemaless, Document Oriented
- No need to configure schema upfront
- No need for slow ALTER TABLE – like operations
- Define mapping (schema) to customize the indexing process
  - Require fields to be of certain type
  - If you want text fields that should not be analyzed

Distributed & Highly Available

- Multiple nodes running in a cluster
  - Acting as a single service
  - Nodes in cluster that store data or nodes that just help in speeding up search queries
- Sharding
  - Indices are sharded (#shards are configurable)
  - Each Shard can have zero or more replicas
    - Replicas on different servers for failover
- Master
  - Automatic Master detection + failover
  - Responsible for distribution/balancing of shards

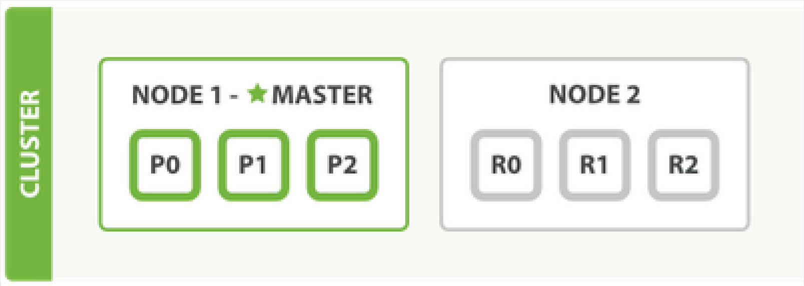
A Single Node Cluster with An Index

- All 3 Primary Shards allocated to Node1
- No replication Nodes
- A single node means single point of failure
- Health of the Cluster: Yellow



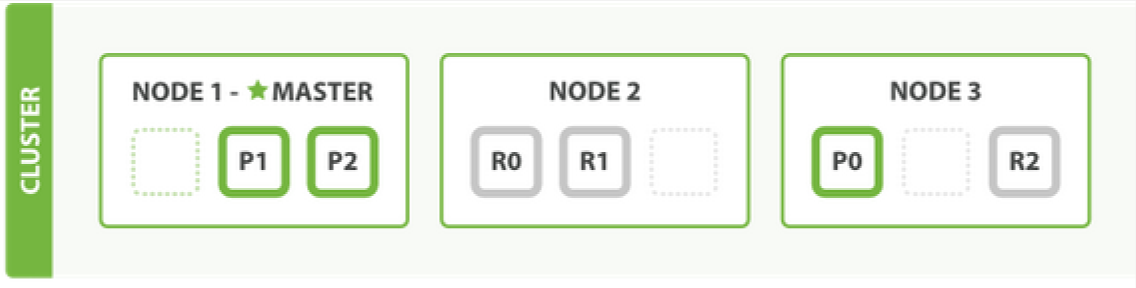
Add Failover

- Add one Node to cluster by configuring the cluster name.
- 3 replica shards have been allocated.
- Cluster Health : Green.
- Now 6 Shards. There is redundancy



Scale horizontally

- A 3 node cluster
- One shard each from Node 1 and Node 2 have moved to Node 3
- Better performance as hardware resouces (CPU,RAM, I/O) are shared

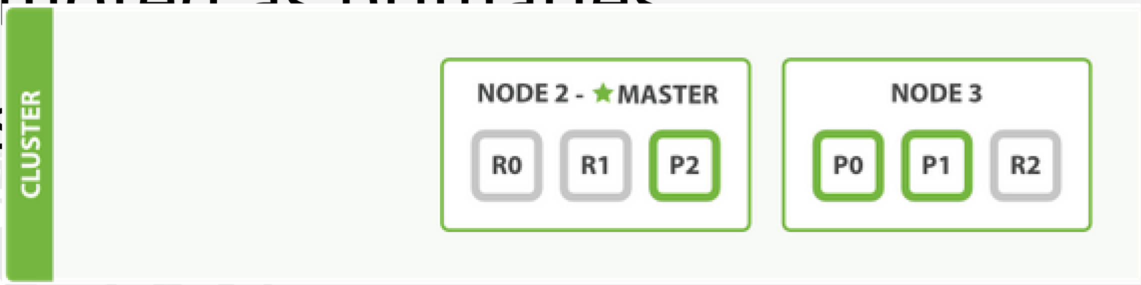


Scale some more

- More Nodes can be added
- More replicas can be added
- This will allow faster searches
- Allows better redundancy
- However the number of **primary shards is fixed** at the moment an index is created.
- Effectively, the maximum amount of data that can be stored in the index is defined by this number.
- Is this a limitation …..

Coping with Node Failure

- Kill Master Node
- Elect a New Master (Node 2)
- Primary Shard 1 and 2 were lost
- Cluster Health : Red
- Node 2 & 3 have Replicas of these shards, which are now promoted as primaries
- Clus



Beauty Of Elastic Search

- In Elasticsearch, **all data in every field** is **indexed by default**. That is, every field has a dedicated inverted index for fast retrieval. And, unlike most other databases, it can use all of those inverted indices **in the same query**, to return results at breathtaking speed