

### **Chapter 1: An Introduction to SMACK**





### Chapter 2: The Model - Scala and Akka

Figure 2-1. The Scala collections top hierarchy.







Figure 2-3- The Map hierarchy





#### **Chapter 3: The Engine - Apache Spark**



Figure 3-1 Apache Spark download page



Figure 3-2 Terminal window with Spark running



Figure 3-3 One driver program with three worker nodes



Figure 3-4. Distributed Spark application

) Spark sl	hell - Spark Jobs 🗙 📃				
⇒ <b>C</b>	C localhost:4040/job	os/			<u>ح</u>
Spa	Jobs	Stages Storage	Environn	nent Executors SQL	Spark shell application U
Spar	k Jobs <sup>(?)</sup>				
Total Uptime: 100.4 h Scheduling Mode: FIFO Completed Jobs: 13 > Event Timeline Completed Jobs (13)					
Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
12	foreach at <console>:24</console>	2016/05/20 03:26:05	15 ms	1/1	4/4
11	foreach at <console>:24</console>	2016/05/20 03:25:35	9 ms	1/1	4/4
10	foreach at <console>:24</console>	2016/05/20 03:25:19	15 ms	1/1	4/4
10 9	foreach at <console>:24 foreach at <console>:24</console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57	15 ms 15 ms	1/1 1/1	4/4
10 9 8	foreach at <console>:24 foreach at <console>:24 take at <console>:24</console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20	15 ms 15 ms 11 ms	1/1 1/1 1/1	4/4 4/4 3/3
10 9 8 7	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24</console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20	15 ms 15 ms 11 ms 6 ms	1/1 1/1 1/1 1/1	4/4 4/4 3/3 1/1
10 9 8 7 5	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24 reduce at <console>:24</console></console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20 2016/05/20 03:23:27	15 ms 15 ms 11 ms 6 ms 10 ms	1/1 1/1 1/1 1/1 1/1	4/4 4/4 3/3 1/1 4/4
10 9 3 7 6 5	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24 reduce at <console>:24 collect at <console>:24</console></console></console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20 2016/05/20 03:23:27 2016/05/20 03:22:40	15 ms 15 ms 11 ms 6 ms 10 ms 7 ms	1/1 1/1 1/1 1/1 1/1 1/1	4/4 3/3 1/1 4/4
10 9 8 7 6 5 4	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24 reduce at <console>:24 collect at <console>:24 count at <console>:24</console></console></console></console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20 2016/05/20 03:23:27 2016/05/20 03:22:24 2016/05/20 03:22:22	15 ms 15 ms 11 ms 6 ms 10 ms 7 ms 16 ms	1/1 1/1 1/1 1/1 1/1 1/1 1/1	4/4 4/4 3/3 1/1 4/4 4/4 4/4
10 9 3 7 5 5 4 3	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24 reduce at <console>:24 collect at <console>:24 count at <console>:24 coulect at <console>:24</console></console></console></console></console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20 2016/05/20 03:23:27 2016/05/20 03:22:24 2016/05/20 03:22:22 2016/05/20 03:08:30	15 ms 15 ms 11 ms 6 ms 10 ms 7 ms 16 ms 0.1 s	1/1 1/1 1/1 1/1 1/1 1/1 1/1 3/3	4/4 4/4 3/3 1/1 4/4 4/4 4/4 12/12
10 9 7 6 5 4 3 2	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24 reduce at <console>:24 collect at <console>:24 count at <console>:24 collect at <console>:24 collect at <console>:26</console></console></console></console></console></console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20 2016/05/20 03:23:27 2016/05/20 03:22:40 2016/05/20 03:22:22 2016/05/20 03:08:30 2016/05/20 03:04:27	15 ms 15 ms 11 ms 6 ms 10 ms 7 ms 16 ms 0.1 s 0.1 s	1/1 1/1 1/1 1/1 1/1 1/1 1/1 3/3 3/3	4/4 4/4 3/3 1/1 4/4 4/4 4/4 12/12 12/12
10 9 7 6 5 4 3 2 1	foreach at <console>:24 foreach at <console>:24 take at <console>:24 take at <console>:24 reduce at <console>:24 collect at <console>:24 count at <console>:24 collect at <console>:24 collect at <console>:24 collect at <console>:26 collect at <console>:26</console></console></console></console></console></console></console></console></console></console></console>	2016/05/20 03:25:19 2016/05/20 03:24:57 2016/05/20 03:24:20 2016/05/20 03:24:20 2016/05/20 03:23:27 2016/05/20 03:22:20 2016/05/20 03:08:30 2016/05/20 03:04:27 2016/05/20 03:00:06	15 ms 15 ms 11 ms 6 ms 10 ms 7 ms 16 ms 0.1 s 0.1 s 0.2 s	1/1 1/1 1/1 1/1 1/1 1/1 1/1 3/3 3/3 3/3	4/4 4/4 3/3 1/1 4/4 4/4 4/4 12/12 12/12 12/12

Figure 3-5. Spark shell application web UI



Figure 3-6. Spark Streaming Operation



Figure 3-7. A DStream as an RDD series



Figure 3-8. Spark Streaming execution with Spark components



Batch Length: 2 seconds, Window Length: 6 seconds, Slide Interval: 4 seconds

Figure 3-9. Windowed operations example

#### **Chapter 4: The Storage - Apache Cassandra**



Figure 4-1 CAP Brewer's theorem



Figure4-2: Column



Figure 4-3: Super column



Figure 4-4: Column family



Figure 4-5: Super column family

Cluster									
KeySpace1				KeySpace2					
ColumnFamily1				ColumnFamily1			ColumnFamily2		
SuperColumn		SuperC	SuperColumn		SuperColumn			SuperColumn	
Column1	Column2	Column1	Column2	Column1	Column2	Column3	Column1	Column2	
Value1	Value2	Value1	Value2	Value1	Value2	Value3	Value1	Value2	

Figure 4-6: Cluster with key spaces



Figure 4-7: Nodes within a cluster



Figure 4-8: DataStax OpsCenter

E Computer Management			
File Action View Help			
🗢 🏟 🖄 📰 🖾 🔒 🛛			
🎥 Computer Management (Local	O. Services		Actions
System Tools			Services 🔺
Task Scheduler	DataStax_Cassandra_Community_Server	Name	More Actions
Event Viewer	Stop the service Restart the service Description:	🖓 Bluetooth Support Service	More Actions P
Shared Folders		SranchCache	DataStax_Cassandra_Comm 🔺
Beformance		🖓 Certificate Propagation	More Actions
Device Manager		🐝 CNG Key Isolation	
4 Ge Storage		🛸 COM+ Event System	
Disk Management	DataStax Cassandra Community Server	🔍 COM+ System Application	
Services and Applications		🖓 Computer Browser	
Services		😪 Credential Manager	
WMI Control		🔍 Cryptographic Services	
		Community_Server	
		🔍 DataStax_OpsCenter_Agent	
		🖗 DataStax_OpsCenter_Community	
		COM Server Process Launcher	
		🔍 Deep Freeze 65 Server Service	
		🔍 Desktop Window Manager Session Manager	
		🔍 DHCP Client	
		🔍 Diagnostic Policy Service	
		Chagnostic Service Host	
		🔍 Diagnostic System Host	
		🔍 Disk Defragmenter	
		🐘 Distributed Link Tracking Client	
		Contraction Coordinator	
		Q DNS Client	
4		Second Se	
		Stensible Authentication Protocol	
		Sa Fax	
		<pre></pre>	
< III >	Extended Standard		-

Figure 4-9: Microsoft Windows display services



Figure 4-10: Display cluster in OpsCenter



Figure 4-11: Apache Cassandra cache



# Chapter 5: The Broker - Apache Kafka

Figure 5-1. Apache Kafka typical scenario

🥖 Apache Kafka 🛛 🗙 📃	Raul – 🗖 🗙
← → C □ kafka.apache.org/dow	nloads.html 😒 🚍
<b>Sea Apa</b> A high-thro	nche Kafka Dughput distributed messaging system.
download introduction	Releases
uses documentation	0.10.0.0 is the latest release. The current stable version is 0.10.0.0.
quickstart performance	You can verify your download by following these <b>procedures</b> and using these <b>KEYS</b> .
clients ecosystem	0.10.0.0
security fag	Release Notes
project	• Source download: kafka-0.10.0.0-src.tgz (asc, md5)
<ul> <li>twitter</li> </ul>	Binary downloads:
• wiki	<ul> <li>Scala 2.10 - kafka_2.10-0.10.0.0.tgz (asc, md5)</li> </ul>
• mailing lists	<ul> <li>Scala 2.11 - kafka_2.11-0.10.0.0.tgz (asc, md5)</li> </ul>
<ul> <li>committers</li> </ul>	We build for multiple versions of Scala. This only matters if you are using Scala and you want a version
<ul> <li>powered by</li> </ul>	built for the same Scala version you use. Otherwise any version should work (2.11 is recommended).

Figure 5-2. Apache Kafka download page



Figure 5-3. Single node - single broker Kafka cluster example



Figure 5-4. Single node - multiple broker Kafka cluster example.



Figure 5-5. Multiple node - multiple broker Kafka cluster



Figure 5-6. A topic with 3 partitions

# Chapter 7: Study Case 1 - Spark and Cassandra



Figure 7-1. Canonical Spark Cassandra cluster



Figure 7-2. Cassandra process and Spark worker one to one relationship



Figure 7-3. Step 1 - Define the business logic



Figure 7-4. Step 2 - Driver send the tasks to the Spark Master





Figure 7-5. Step 3 - The Spark Master distributes the task among the workers



Figure 7-6. Step 4 - The Spark worker executes the task with the Spark executor



Figure 7-7. Step 5 - The Spark Executor executes the task with the Cassandra Process



Figure 7-8. Data Locality



Figure 7-9. Read data from Cassandra



Figure 7-10. Spark shuffle operations

# Async batches fan-out writes to Cassandra



Figure 7-11. Async writes to Cassandra (without data locality)

rdd.repartitionByCassandraReplica("keyspace","table")

Write to Cassandra



Figure 7-12. Write to Cassandra with Data Locality







Figure 7-14. Spark Cassandra use cases

# **Chapter 8: Study Case 2 – Connectors**



Figure 8-1. Twitter downloader actors

### Chapter 9: Study Case 3 - Mesos and Docker



Virtual Machine

Docker Container

Figure 9.1. Comparison between a virtual machine and a Docker container



Figure 9.2. Containerization in Mesos