The Apache Cassandra storage engine

Sylvain Lebresne

NoSQL matters 2012
About me

- Sylvain Lebresne
- sylvain@datastax.com
- @pcmanus
1. What is Apache Cassandra

2. Data Model

3. The storage engine
1. What is Apache Cassandra
2. Data Model
3. The storage engine
about:project

• Distributed data store aimed at big data.
• Apache project since 2010.
• Version 1.1 released last month.
• Proven in production (Netflix, Twitter, Reddit, Cisco, ...). Largest know cluster has over 300TB in over 400 machines.
Apache Cassandra
Apache Cassandra

A database:
Apache Cassandra

A database:
- distributed / decentralized
Apache Cassandra

A database:
- distributed / decentralized
- replicated & durable
Apache Cassandra

A database:
• distributed / decentralized
• replicated & durable
• scalable / elastic
Apache Cassandra

A database:
- distributed / decentralized
- replicated & durable
- scalable / elastic
Apache Cassandra

A database:
- distributed / decentralized
- replicated & durable
- scalable / elastic
- fault-tolerant / no SPOF
Apache Cassandra

A database:
• distributed / decentralized
• replicated & durable
• scalable / elastic
• fault-tolerant / no SPOF
• highly available
Apache Cassandra

A database:
- distributed / decentralized
- replicated & durable
- scalable / elastic
- fault-tolerant / no SPOF
- highly available
- data center aware
1. What is Apache Cassandra

2. Data Model

3. The storage engine
Data Model

• Not SQL (no transaction, nor joins) but more than Key/Value.
• Inspired by Google BigTable
• Column families based.
Ex: user profiles

“For each user, holds profile infos”

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>50e8-e29b</td>
</tr>
<tr>
<td>birth_year</td>
<td>1994</td>
</tr>
<tr>
<td>fname</td>
<td>Justin</td>
</tr>
<tr>
<td>lname</td>
<td>Bieber</td>
</tr>
</tbody>
</table>
Ex: user profiles

“For each user, holds profile infos”

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>50e8-e29b</td>
<td></td>
</tr>
<tr>
<td>birth_year</td>
<td>1994</td>
</tr>
<tr>
<td>fname</td>
<td>Justin</td>
</tr>
<tr>
<td>lname</td>
<td>Bieber</td>
</tr>
<tr>
<td>2ab1-f1b7</td>
<td></td>
</tr>
<tr>
<td>birth_year</td>
<td>1978</td>
</tr>
<tr>
<td>email</td>
<td><a href="mailto:a@kutcher.com">a@kutcher.com</a></td>
</tr>
<tr>
<td>fname</td>
<td>Ashton</td>
</tr>
<tr>
<td>lname</td>
<td>Kutcher</td>
</tr>
</tbody>
</table>
Ex: user’s Tweets

“For each user, tweets he has made”
Ex: user’s Tweets

“For each user, tweets he has made”

Timeline

50e8-e29b

0

@LiveLoveKary glad you had a good birthday #muchlove
Ex: user’s Tweets

“For each user, tweets he has made”

50e8-e29b

1
@NickDeMoura happy bday my dude.

0
@LiveLoveKary glad you had a good birthday #muchlove
Ex: user’s Tweets

“For each user, tweets he has made”

```
50e8-e29b

2
@MickyArison @miamiHEAT thanks for the gam tonight

1
@NickDeMoura happy bday my dude.

0
@LiveLoveKary glad you had a good birthday #muchlove
```
Ex: user’s Tweets

“For each user, tweets he has made”

50e8-e29b

3  still a little tired. back in the studio today with Timbaland
2  @MickyArison @miamiHEAT thanks for the gam tonight
1  @NickDeMoura happy bday my dude.
0  @LiveLoveKary glad you had a good birthday #muchlove
There’s more

• Secondary indexes
• Distributed counters
• Composite columns
1. What is Apache Cassandra

2. Data Model

3. The storage engine
Goal

• Writes are harder than reads to scale
• Spinning disks aren’t good with random I/O
• Goal: minimize random I/O
A write’s journey

write((k1, v1))
A write’s journey

write((k1, c1:v1))

Memory

Memtable

Commit log

Hard drive
A write’s journey
A write’s journey

write(k2, c1:v1 c2:v2)

Memory

Hard drive
A write’s journey

write((k1, c1:v4), (c3:v3, c2:v2))
A write’s journey
A write’s journey

more updates

Memory

Hard drive

index
A write’s journey

Memory

flush

index

Hard drive

k1 c1:v4 c2:v2 c3:v3
k2 c1:v1 c2:v2

index

k1 c1:v5 c4:v4
k2 c1:v2 c3:v3
Writes properties

- No reads or seeks
- Only sequential I/O
- Immutable SSTables: easy snapshots
A read’s journey

read(k1)

Memory

Hard drive

index

k1: c1:v4, c2:v2, c3:v3
k2: c1:v1, c2:v2

index

k1: c1:v5, c2:v4
k2: c1:v2, c3:v3
A read’s journey

[Diagram showing a read's journey through memory and hard drive with key-value pairs and merges.]
Compaction

• Goal: keep the number of SSTables low
• Merge sort against multiple sstables
• Sequential I/O
Compaction

- Goal: keep the number of SSTables low
- Merge sort against multiple sstables
- Sequential I/O
Compaction

- Goal: keep the number of SSTables low
- Merge sort against multiple sstables
- Sequential I/O
SSTables
Optimizations

• Row Cache
• Bloom filters: eliminates whole SSTable
• Key Cache
• Rows & Columns Indexes
• ...

©2012 DataStax
Other features

- Compression
- Checksums
- Time to live
QUESTIONS?
• http://cassandra.apache.org/
• http://wiki.apache.org/cassandra/
• http://www.datastax.com/docs/1.0