



Cassandra & Redis

DIEGO PACHECO

About me...



- Cat's Father
- Principal Software Architect
- Agile Coach
- SOA/Microservices Expert
- DevOps Practitioner
- Speaker
- Author

 diegopacheco

 @diego_pacheco

 <http://diego-pacheco.blogspot.com.br/>



<https://diegopacheco.github.io/>



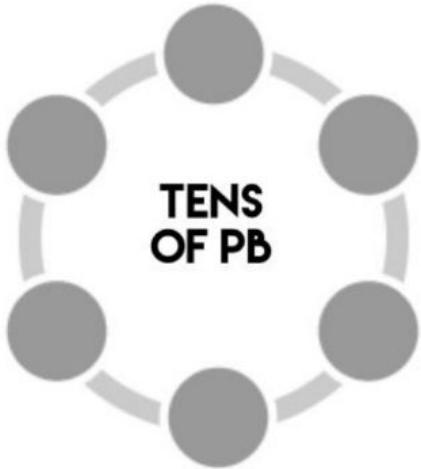
Cassandra

Cassandra

- FOSS
- NoSQL Database
- Low Latency
- High Throughput
- Battle tested by Netflix, Apple, Facebook.
- AP on CAP - Favors availability over consistency.
- Multi-region support.
- Great software but hard to operate.



Cassandra Battle Tested



NETFLIX



Spotify

Cassandra ~ Netflix 2013

Operational Simplicity June 2013 

33M streaming customers

2T API calls/year

~1,200 Servers

55 AWS clusters

12 developers

4 operators

0 New data centers

**“Our primary operational data store
is now Cassandra, not Oracle.”**



Cassandra ~ Apple 2016

Massive

- Cassandra Nodes: 75,000+
- Data Size: 10s of PetaBytes
- Ops/sec: In the millions
- Largest Cluster: 1000+
- Versions: 1.2.x and 2.0.x



Cassandra ~ Spotify 2019

Cassandra at Spotify



100+
production clusters



3000+
production machines



150+ TB
of unique data



4 million
queries per second
at peak



Cassandra ~ Data Model

Data Model



| Cassandra | RDBMS Equivalent |
|------------------|------------------|
| KEYSPACE | DATABASE/SCHEMA |
| COLUMN FAMILY | TABLE |
| ROW | ROW |
| FLEXIBLE COLUMNS | DEFINED COLUMNS |

How to have Success with Cassandra?



- ❑ Understand system / service / app requirements
- ❑ Identify your data access patterns
- ❑ Model around the access patterns
- ❑ Denormalization is your friend (by^{*} query tables)
- ❑ Benchmark
- ❑ Tuneup
- ❑ Repeat

Netflix Benchmark: 1M writes/sec



Benchmark: 1 million writes per second

Per Node Activity

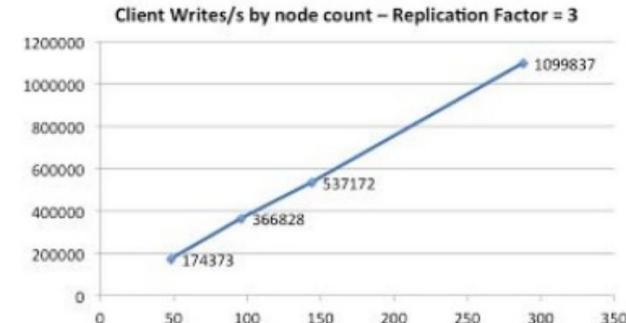
| Per Node | 48 Nodes | 96 Nodes | 144 Nodes | 288 Nodes |
|---------------------|-------------|-------------|-------------|-------------|
| Per Server Writes/s | 10,900 w/s | 11,460 w/s | 11,900 w/s | 11,456 w/s |
| Mean Server Latency | 0.0117 ms | 0.0134 ms | 0.0148 ms | 0.0139 ms |
| Mean CPU %Busy | 74.4 % | 75.4 % | 72.5 % | 81.5 % |
| Disk Read | 5,600 KB/s | 4,590 KB/s | 4,060 KB/s | 4,280 KB/s |
| Disk Write | 12,800 KB/s | 11,590 KB/s | 10,380 KB/s | 10,080 KB/s |
| Network Read | 22,460 KB/s | 23,610 KB/s | 21,390 KB/s | 23,640 KB/s |
| Network Write | 18,600 KB/s | 19,600 KB/s | 17,810 KB/s | 19,770 KB/s |

Node specification – Xen Virtual Images, AWS US East, three zones

- Cassandra 0.8.6, CentOS, SunJDK6
- AWS EC2 m1 Extra Large – Standard price \$ 0.68/Hour
- 15 GB RAM, 4 Cores, 1Gbit network
- 4 internal disks (total 1.6TB, striped together, md, XFS)

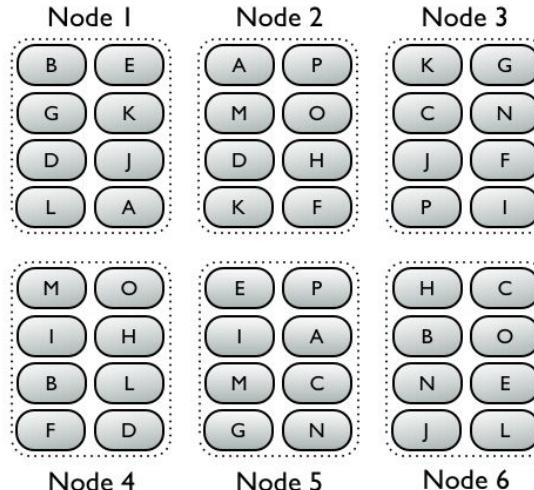
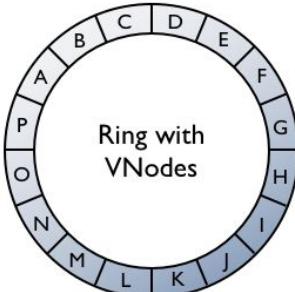
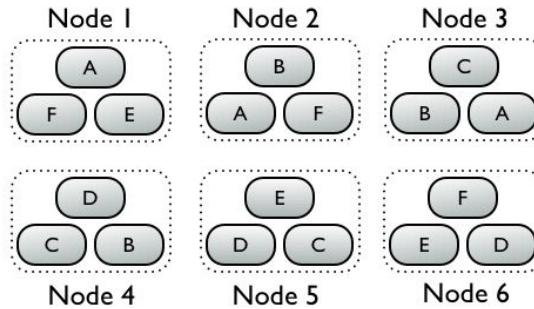
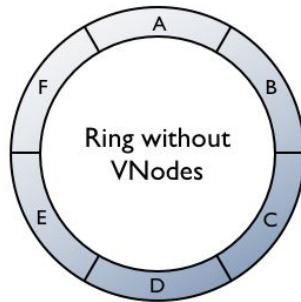


Scale-Up Linearity



<http://techblog.netflix.com/2011/11/benchmarking-cassandra-scalability-on.html>

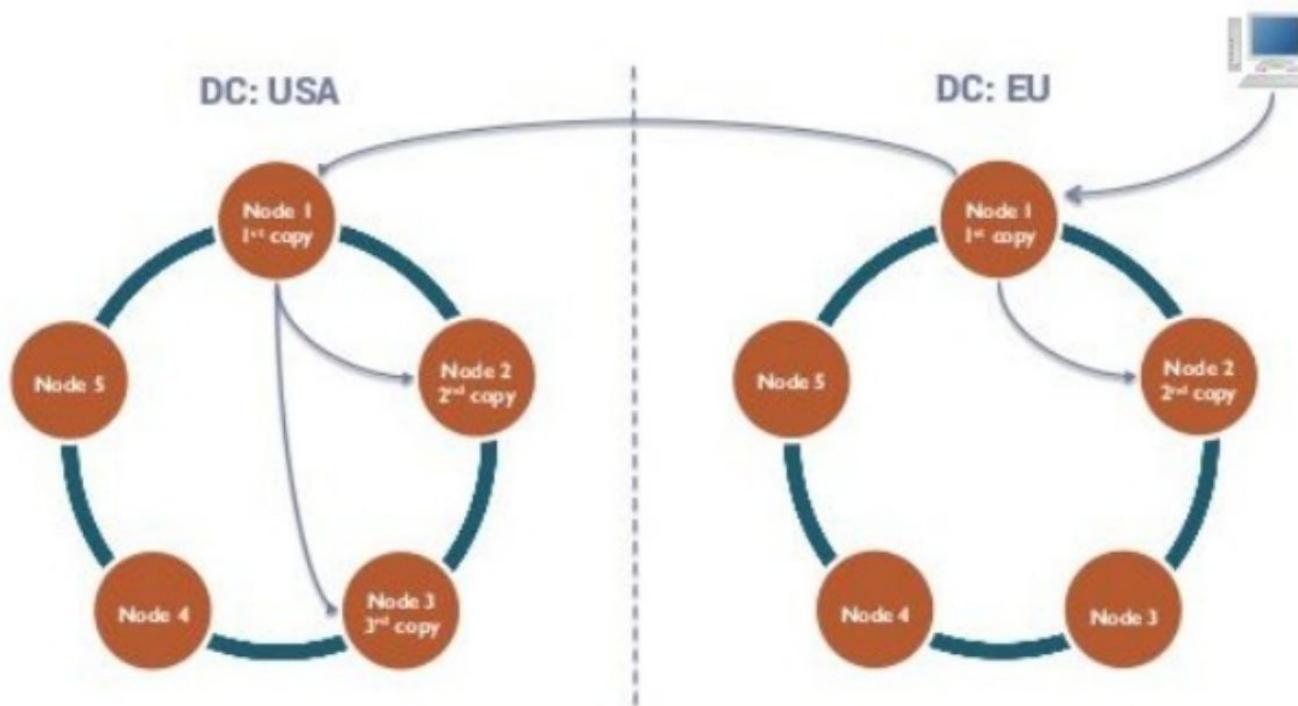
Cassandra Cluster



- Master less
- Anti-entropy HH
- Consistent Hashing
- Murmur3
- Virtual Nodes
- Dynamic Scale out

Cassandra Data Replication

```
CREATE KEYSPACE johnny WITH REPLICATION =  
{'class': 'NetworkTopologyStrategy', 'USA':3, 'EU': 2};
```



Cassandra Tunable Consistency



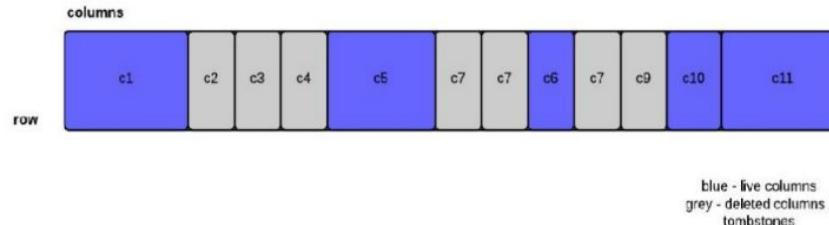
- ❑ Tunable for Read / Writes
- ❑ Consistency VS Availability
 - ❑ One, Two, Three,
 - ❑ QUORUM(majority $n/2+1$), Local_Quorum
 - ❑ Each_Quorum
 - ❑ Local_One
 - ❑ All, Any

Cassandra Tombstones

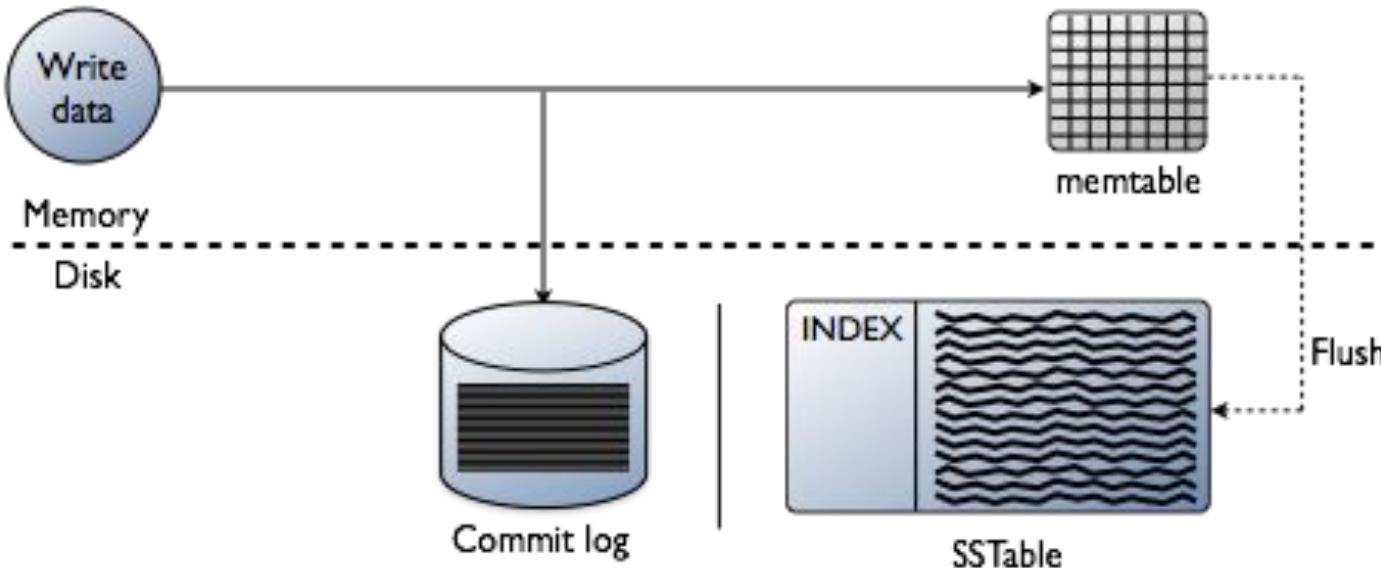
Tombstones



- ❑ Deleted data is MARKED as Removed == Tombstone
- ❑ Data is deleted and removed during compaction
- ❑ Compaction can happen in few days depending of the configs.
- ❑ Queries on partition with lots of tombstones requires lots of filtering which can slow down the CASS performance.
- ❑ Collections operations can lead to tombstones depending on what you do.
- ❑ There are Compaction Trade-Offs.



Cassandra: Write Path



Cassandra: Read Path

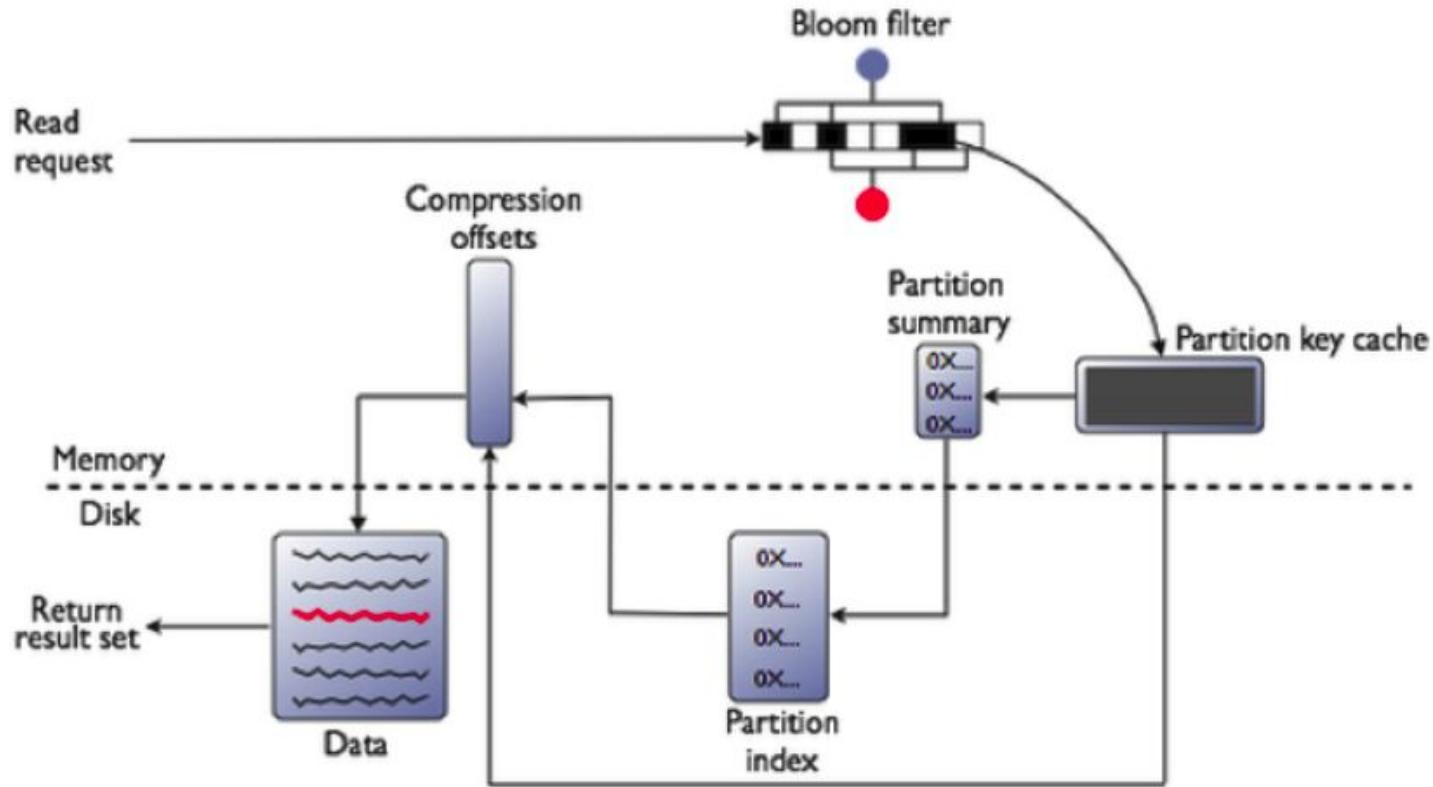


Figure 3. The Cassandra Read Path

Cassandra: Reads & Index



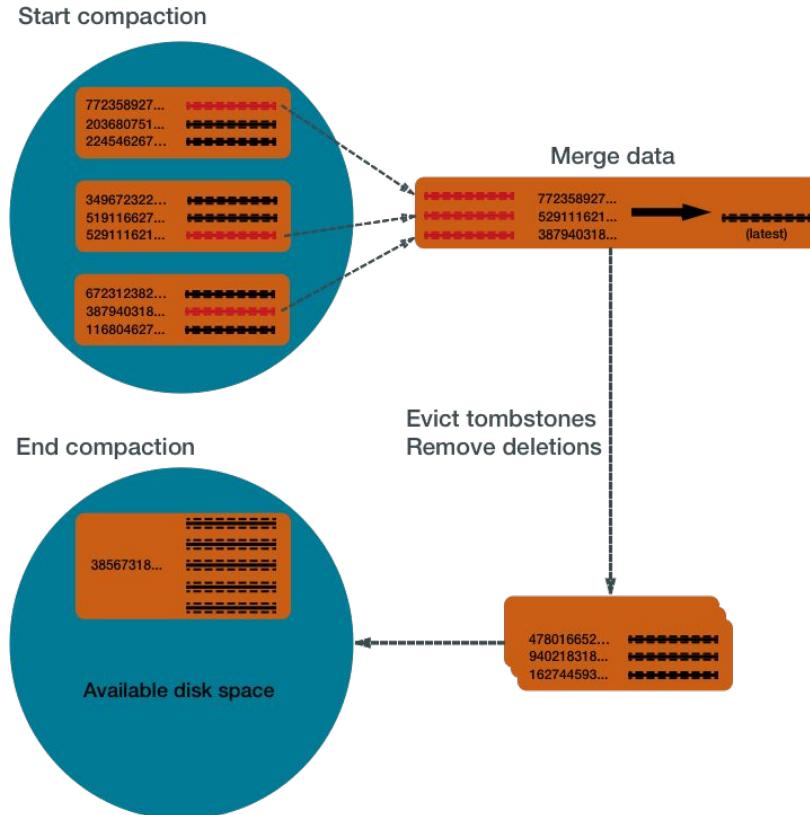
- ❑ Partition Key Cache / Row Cache
 - ❑ Off Heap
 - ❑ Configurable
- ❑ Secondary Index
 - ❑ Filters data on table by non-primary key
 - ❑ Allow Filtering (it can be problematic)

Cassandra: Compaction

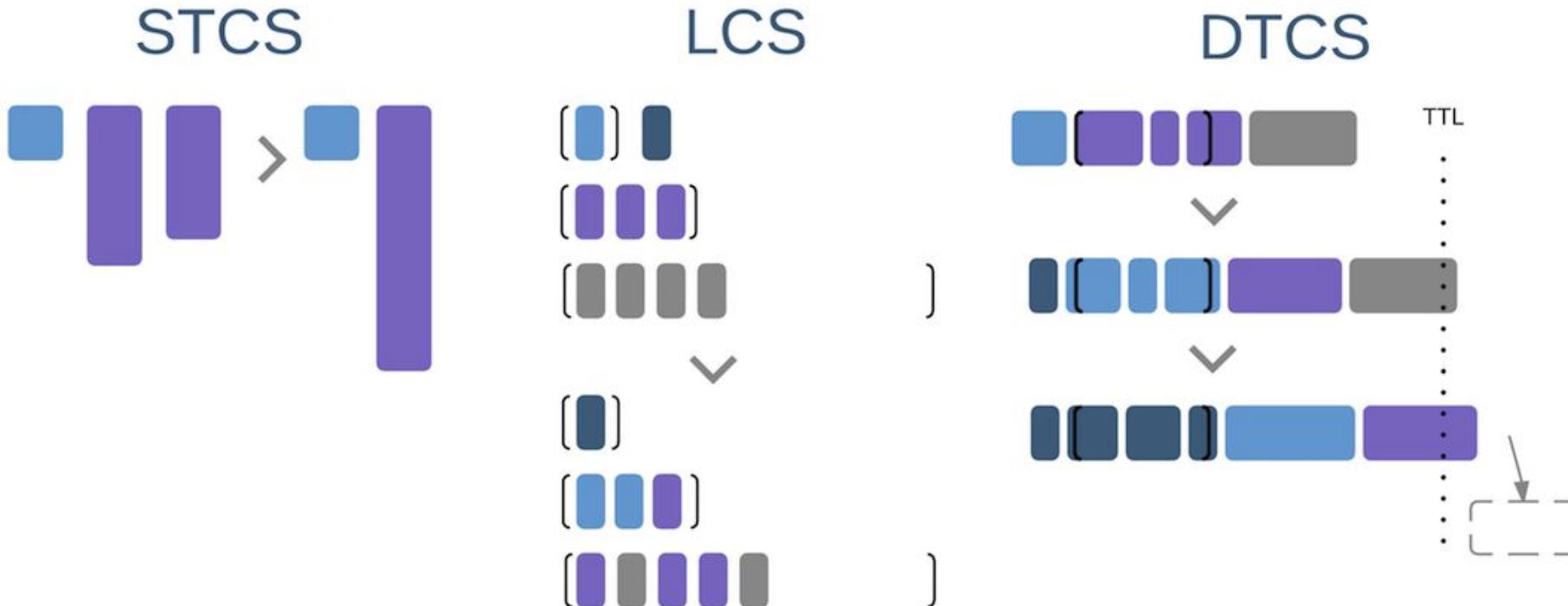


- ❑ Merge SSTables
- ❑ Can be triggered by node_tool
- ❑ Impact Cassandra Performance
- ❑ Different Strategies for different workloads

Cassandra: Compaction



Cassandra: Compaction Strategies



Cassandra: Repair



- ❑ Anti-entropy mechanism (expensive, slow)
- ❑ 100% needed in Multi-region clusters
- ❑ Other scenarios that you need repairs:
 - ❑ Low consistent reads and writes
 - ❑ Frequent nodes outage
 - ❑ Low reads repair chance
 - ❑ Flaky networks

Cassandra: Read Repair



- ❑ Easy and cheap (when reading from multiples replicas)
- ❑ Hot Data will be repaired a lot

Cassandra: When to not Repair

- ❑ Short TTLs
- ❑ High Consistent read and writes
- ❑ High Read repair chance



Cassandra: Repair Pain points - most on scale



- ❑ Stream timeouts
- ❑ Tracking / Resume Repairs
- ❑ CPU Usage
- ❑ Disk I/O
- ❑ Latencies / SLA
- ❑ Wide Partitions

Cassandra: Poor Usage - Anti-Patterns



- ❑ Batch usage: Select * from whole table / re-insert periodically
- ❑ Lack of denormalized tables to support queries (allow filtering)
- ❑ Queue
- ❑ Extreme large row sizes
- ❑ Abuse of Collections Data types(tombstones)
- ❑ Read Before Writing (or write and read immediately.)

Running Cassandra: \$ bin/cassandra



```
diego@4winds: ~/bin/apache-cassandra-3.11.5
File Edit View Search Terminal Tabs Help
diego@4winds: ~/bin/apache-cassandra-3.11.5          bin/cqlsh
1, -7493286240810977657, -7494544754733097624, -7516071104531496729, -7519337364151595194, -7528252338206538254, -7574796766262570422, -780422736851399801, -7846193786383469867, -7860249652509063110, -8030586380928297394, -8100418713550062970, -8197782966774991959, -8206592406781134180, -8252267470609826869, -8255989648356626466, -8416580432061226637, -8446733811771566437, -846582169894377500, -8582918576196693584, -8639847981812625078, -8700198368396344080, -8735921828659138179, -8788629091745656086, -8891882710544106984, -895942689336303593, -8973287893808580595, -904562398326852580, -908217712635510494, -9089041005318193567, -9095007526468268387, -9171213106701495137, 118410192309618278, 1322306179201640260, 1330692058419493135, 140567342758702722, 1430554436327072748, 1469661973386823839, 1483367716586491706, 161722331140746179, 1639125997569239284, 1650564288459786088, 166357870049935329, 1709750776869570724, 1754733125345366290, 1975339127888333360, 2042703901774115410, 2564224747046821959, 2611601529161595021, 2711007235424858912, 2737574029600407655, 2796146810959629825, 2825394653478931797, 2851676156548594296, 2897681460382091807, 2950684283781668979, 2993436814147875483, 3109215840984061072, 3162070349659144815, 3231871418246665040, 3329640069918439972, 334404898982155234, 345251057145315557, 3536790151939085764, 36949549461048436162, 3765154151544455794, 383165265528614193, 3943636890349815808, 3991687307211180174, 3998468200772060008, 4045726092026436715, 4063799096289946037, 419890407008810358, 4222770610308327099, 4273859191943886652, 4373287450246892074, 459109098619520290, 4594877426033211868, 4609136865449109029, 4644272517100885113, 4646358172463262215, 4763264414133828539, 4866259494898323591, 4962963159849081768, 4975225162916489177, 5065611366142192280, 5297792356099254626, 5325451566932147787, 5349313779953435279, 5365355694924613558, 5513898114273064153, 5581884114693605706, 5584200697738392089, 5619840213323209180, 5661142774010017782, 566610601385947792, 5700812505277903524, 5750851064339175569, 5851109862498950029, 5854338266548559383, 590494209373732056, 6000044287739472607, 6028903157590339429, 6118437714400035883, 6205798363891767018, 624493612981469346, 6262968242161817791, 6393306617227077352, 6394485496142540874, 6468124582867741755, 648268215788755861, 6492621395891371667, 6634966314212061489, 6664608487782343452, 6672086457070682433, 6733445328625964759, 6813232966366766951, 6830764375232354297, 6945358248406253967, 697908347231986977, 7113129868221166861, 7145601124743867668, 71551766651257277, 7273527555058861296, 7354414917705148950, 7371828536145294022, 7398940567907282305, 7413330900565281715, 7421056578546218467, 7431806640376616653, 7446617571951179837, 7469579454391301306, 748210969631533088, 7513638648008425680, 7649707905590851218, 7671249080036170028, 7701622584610576130, 7798705419131445476, 782802815184559756, 7878905355564837325, 8074841591006367753, 8106886858864220375, 8132213678513289560, 8162381542610387068, 8171959606319217935, 8208105456825855356, 8226335692449099276, 8288944860482662994, 8424501362844050226, 8432436616338635593, 853299401074508704, 8581078177871972574, 8643318304053629967, 8721829069000446464, 8739870853929027548, 8831819811070284367, 8836042284068804020, 8990944083847987631, 8996333603836965182, 9068324662939877137, 9072915054691135892, 9155859756012079814, 9171075937585297745, 9204322407704970925, 931367151313224227, 932311335709728507, 978857672887964909, 989690790672155715]INFO [main] 2019-11-21 16:48:15,971 StorageService.java:1521 - JOINING: Finish joining ringINFO [main] 2019-11-21 16:48:16,160 StorageService.java:2442 - Node localhost/127.0.0.1 state jump to NORMAL
```

Running Cassandra: \$ bin/cqlsh

```
bin/cqlsh
File Edit View Search Terminal Tabs Help
diego@4winds: ~/bin/apache-cassandra-3.11.5
diego@4winds ~ ~/bin/apache-cassandra-3.11.5 bin/cqlsh
bin/cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.5 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> █
```



Running Cassandra: Java Code

```
1 import com.datastax.driver.core.Cluster;
2 import com.datastax.driver.core.ResultSet;
3 import com.datastax.driver.core.Row;
4 import com.datastax.driver.core.Session;
5
6 public class Basic {
7     Run | Debug
8     public static void main(String[] args) {
9
10         Cluster cluster = null;
11         try {
12             cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
13             Session session = cluster.connect();
14             ResultSet rs = session.execute("select release_version from system.local");
15             Row row = rs.one();
16             System.out.println(row.getString("release_version"));
17         } finally {
18             if (cluster != null)
19                 cluster.close();
20         }
21     }
22 }
```



Running Cassandra: Java Code



```
nov 21, 2019 5:02:36 PM com.datastax.driver.core.Cluster logDriverVersion
INFO: DataStax Java driver 3.8.0 for Apache Cassandra
nov 21, 2019 5:02:36 PM com.datastax.driver.core.GuavaCompatibility selectImplementation
INFO: Detected Guava >= 19 in the classpath, using modern compatibility layer
nov 21, 2019 5:02:37 PM com.datastax.driver.core.ClockFactory newInstance
INFO: Using native clock to generate timestamps.
nov 21, 2019 5:02:37 PM com.datastax.driver.core.NettyUtil <clinit>
INFO: Did not find Netty's native epoll transport in the classpath, defaulting to NIO.
nov 21, 2019 5:02:37 PM com.datastax.driver.core.policies.DCAwareRoundRobinPolicy init
INFO: Using data-center name 'datacenter1' for DCAwareRoundRobinPolicy (if this is incorrect, please provide the correct
datacenter name with DCAwareRoundRobinPolicy constructor)
nov 21, 2019 5:02:37 PM com.datastax.driver.core.Cluster$Manager init
INFO: New Cassandra host /127.0.0.1:9042 added
3.11.5
```

Running Cassandra: Java Code



```
src > main > java > ① SchemaQuery.java > ④ SchemaQuery > ⑤ main(String[])
 1  import com.datastax.driver.core.Cluster;
 2  import com.datastax.driver.core.ResultSet;
 3  import com.datastax.driver.core.Row;
 4  import com.datastax.driver.core.Session;
 5
 6  public class SchemaQuery {
 7      Run|Debug
 8      public static void main(String[] args) {
 9          Cluster cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
10          Session session = cluster.connect();
11          try {
12              session.execute("CREATE KEYSPACE cass_training " +
13                  "WITH replication = { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };");
14              System.out.println("Schema Created");
15          }catch(Exception e){
16          }
17
18          try {
19              session.execute("USE cass_training;");
20              session.execute("CREATE TABLE user " +
21                  " + (fullName text PRIMARY KEY, email text);");
22              System.out.println("Table Created");
23          }catch(Exception e){
24              System.out.println(e);
25          }
26      }
27  }
```

Running Cassandra: Java Code

```
26      try {
27          session.execute("INSERT INTO user " +
28              "(fullName,email) VALUES ('Diego Pacheco','diego.pacheco@gmail.com');");
29          System.out.println("row inserted.");
30      }catch(Exception e){
31          System.out.println(e);
32      }
33
34      try {
35          System.out.println("List data");
36          ResultSet rs = session.execute("SELECT * from user;");
37          for (Row row : rs) {
38              System.out.println("Name: " + row.getString("fullName"));
39              System.out.println("Mail: " + row.getString("email"));
40          }
41          System.out.println("row inserted.");
42      }catch(Exception e){
43          System.out.println(e);
44      }
45
46      if (cluster != null)
47          cluster.close();
48  }
```



Running Cassandra: Java Code



```
File Edit View Search Terminal Tabs Help  
bin/cqlsh x  
cqlsh:cass_training> select * from user;  
  
 fullname | email  
-----+  
 Diego Pacheco | diego.pacheco@gmail.com  
  
(1 rows)  
cqlsh:cass_training> █
```

Running Cassandra: Java Code Async



```
1 import com.datastax.driver.core.Cluster;
2 import com.datastax.driver.core.ResultSet;
3 import com.datastax.driver.core.Session;
4 import com.google.common.base.Function;
5 import com.google.common.util.concurrent.AsyncFunction;
6 import com.google.common.util.concurrent.FutureCallback;
7 import com.google.common.util.concurrent.Futures;
8 import com.google.common.util.concurrent.ListenableFuture;
9
10 public class Async{
11     Run | Debug
12     public static void main(String[] args) throws Exception {
13         Cluster cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
14         ListenableFuture<Session> session = cluster.connectAsync();
15
16         ListenableFuture<ResultSet> resultSet = Futures.transformAsync(session,
17             new AsyncFunction<Session, ResultSet>() {
18                 public ListenableFuture<ResultSet> apply(Session session) throws Exception {
19                     return session.executeAsync("select release_version from system.local");
20                 }
21             });
22     }
23 }
```

Running Cassandra: Java Code Async



```
22
23     ListenableFuture<String> version = Futures.transform(resultSet,
24     new Function<ResultSet, String>() {
25         public String apply(ResultSet rs) {
26             return rs.one().getString("release_version");
27         }
28     });
29
30     Futures.addCallback(version, new FutureCallback<String>() {
31         public void onSuccess(String version) {
32             System.out.printf("Cassandra version: %s%n", version);
33         }
34         public void onFailure(Throwable t) {
35             System.out.printf("Failed to retrieve the version: %s%n",
36                             t.getMessage());
37         }
38     });
39
40     Thread.sleep(3000L);
41     if (cluster != null)
42         cluster.close();
43 }
44 }
```

Running Cassandra: Java Code Async



```
nov 22, 2019 1:34:35 PM com.datastax.driver.core.Cluster logDriverVersion
INFO: DataStax Java driver 3.8.0 for Apache Cassandra
nov 22, 2019 1:34:35 PM com.datastax.driver.core.GuavaCompatibility selectImplementation
INFO: Detected Guava >= 19 in the classpath, using modern compatibility layer
nov 22, 2019 1:34:35 PM com.datastax.driver.core.ClockFactory newInstance
INFO: Using native clock to generate timestamps.
nov 22, 2019 1:34:35 PM com.datastax.driver.core.NettyUtil <clinit>
INFO: Did not find Netty's native epoll transport in the classpath, defaulting to NIO.
nov 22, 2019 1:34:36 PM com.datastax.driver.core.policies.DCAwareRoundRobinPolicy init
INFO: Using data-center name 'datacenter1' for DCAwareRoundRobinPolicy (if this is incorrect, please provide the correct
datacenter name with DCAwareRoundRobinPolicy constructor)
nov 22, 2019 1:34:36 PM com.datastax.driver.core.Cluster$Manager init
INFO: New Cassandra host /127.0.0.1:9042 added
Cassandra version: 3.11.5
```

Running Cassandra: Java Code Mapper

```
2  
3 import com.datastax.driver.mapping.annotations.Column;  
4 import com.datastax.driver.mapping.annotations.PartitionKey;  
5 import com.datastax.driver.mapping.annotations.Table;  
6  
7 @Table(keyspace = "cass_training", name = "user",  
8         readConsistency = "QUORUM",  
9         writeConsistency = "QUORUM",  
10        caseSensitiveKeyspace = false,  
11        caseSensitiveTable = false)  
12 public class User{  
13  
14     @PartitionKey  
15     @Column(name = "fullName")  
16     private String fullName;  
17     private String email;  
18  
19     public User(){}
20  
21     public User(String name, String email){  
22         this.fullName = name;  
23         this.email = email;
24     }
25  
26     public void setEmail(String email) {
27         this.email = email;
28     }
29     public String getEmail() {
30         return email;
31     }
32     public void setFullName(String fullName) {
33         this.fullName = fullName;
34     }
35     public String getFullName() {
36         return fullName;
37     }
38  
39     @Override
40     public String toString() {
41         return "Name: " + this.fullName + " Email: " + this.email;
42     }
43 }
```

```
29     public String getEmail() {
30         return email;
31     }
32     public void setFullName(String fullName) {
33         this.fullName = fullName;
34     }
35     public String getFullName() {
36         return fullName;
37     }
38  
39     @Override
40     public String toString() {
41         return "Name: " + this.fullName + " Email: " + this.email;
42     }
43 }
```



Running Cassandra: Java Code Mapper

```
3  import com.datastax.driver.core.Cluster;
4  import com.datastax.driver.core.ResultSet;
5  import com.datastax.driver.core.Session;
6  import com.datastax.driver.mapping.Mapper;
7  import com.datastax.driver.mapping.MappingManager;
8  import com.datastax.driver.mapping.Result;
9
10 public class MapperMain {
11     Run|Debug
12     public static void main(String[] args) {
13         Cluster cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
14         Session session = cluster.connect();
15
16         MappingManager manager = new MappingManager(session);
17         Mapper<User> mapper = manager.mapper(User.class);
18
19         String name = "" + System.currentTimeMillis();
20         User u = new User("John Cat N" + name, "cat" + name + "@cats.com");
21         mapper.save(u);
22
23         u = mapper.get("Diego Pacheco");
24         System.out.println("user: " + u);
25
26         System.out.println("Getting all users... ");
27         ResultSet results = session.execute("SELECT * FROM cass_training.user");
28         Result<User> users = mapper.map(results);
29         for (User user : users) {
30             System.out.println("User : " + user);
31         }
32     }
}
```



Running Cassandra: Java Code Mapper



```
ncoding=UTF-8 -cp /tmp/cp_6s9o6nzql4yhtqbb6n7o05fq.jar mapper.MapperMain
nov 22, 2019 1:48:54 PM com.datastax.driver.core.Cluster logDriverVersion
INFO: DataStax Java driver 3.8.0 for Apache Cassandra
nov 22, 2019 1:48:54 PM com.datastax.driver.core.GuavaCompatibility selectImplementation
INFO: Detected Guava >= 19 in the classpath, using modern compatibility layer
nov 22, 2019 1:48:54 PM com.datastax.driver.core.ClockFactory newInstance
INFO: Using native clock to generate timestamps.
nov 22, 2019 1:48:54 PM com.datastax.driver.core.NettyUtil <clinit>
INFO: Did not find Netty's native epoll transport in the classpath, defaulting to NIO.
nov 22, 2019 1:48:55 PM com.datastax.driver.core.policies.DCAwareRoundRobinPolicy init
INFO: Using data-center name 'datacenter1' for DCAwareRoundRobinPolicy (if this is incorrect, please provide the correct datacenter name with DCAwareRoundRobinPolicy constructor)
nov 22, 2019 1:48:55 PM com.datastax.driver.core.Cluster$Manager init
INFO: New Cassandra host /127.0.0.1:9042 added
user: Name: Diego Pacheco Email: diego.pacheco@gmail.com
Getting all users...
User : Name: John Cat N1574459233432 Email: cat1574459233432@cats.com
User : Name: Diego Pacheco Email: diego.pacheco@gmail.com
User : Name: John Cat N1574459335562 Email: cat1574459335562@cats.com
User : Name: John Cat N1574459308465 Email: cat1574459308465@cats.com
[]
```

Want to join the DevOps side of the Force?

Learn & Play Local - diegopacheco/cassandra-docker

The screenshot shows the GitHub repository page for `diegopacheco/cassandra-docker`. The page includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and user profile icons. The repository name is displayed prominently, along with a description: "cassandra-docker: Cassandra cluster with Docker". Below the description are buttons for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. A summary bar shows 71 commits, 1 branch, 6 releases, 1 contributor, and an Unlicense. A "Clone or download" button is visible. The commit history lists files like .gitignore, Dockerfile, LICENSE, README.md, cassandra-2.1.19.yaml, cassandra-3.9.yaml, cassandra-docker-mac.sh, cassandra-docker.sh, cassandra-manager.sh, and start-cass.sh, all contributed by diegopacheco, with the latest commit being b435126 on Feb 8.

| File | Description | Time Ago |
|-------------------------|----------------------------------|---------------|
| .gitignore | default version support | a year ago |
| Dockerfile | . | 11 months ago |
| LICENSE | Initial commit | a year ago |
| README.md | . | 11 months ago |
| cassandra-2.1.19.yaml | Working with cass 2.1.19 and 3.9 | a year ago |
| cassandra-3.9.yaml | Working with cass 2.1.19 and 3.9 | a year ago |
| cassandra-docker-mac.sh | default version support | a year ago |
| cassandra-docker.sh | improve docs | 11 months ago |
| cassandra-manager.sh | . | 11 months ago |
| start-cass.sh | Working with cass 2.1.19 and 3.9 | a year ago |

<https://github.com/diegopacheco/cassandra-docker>



Redis

Redis

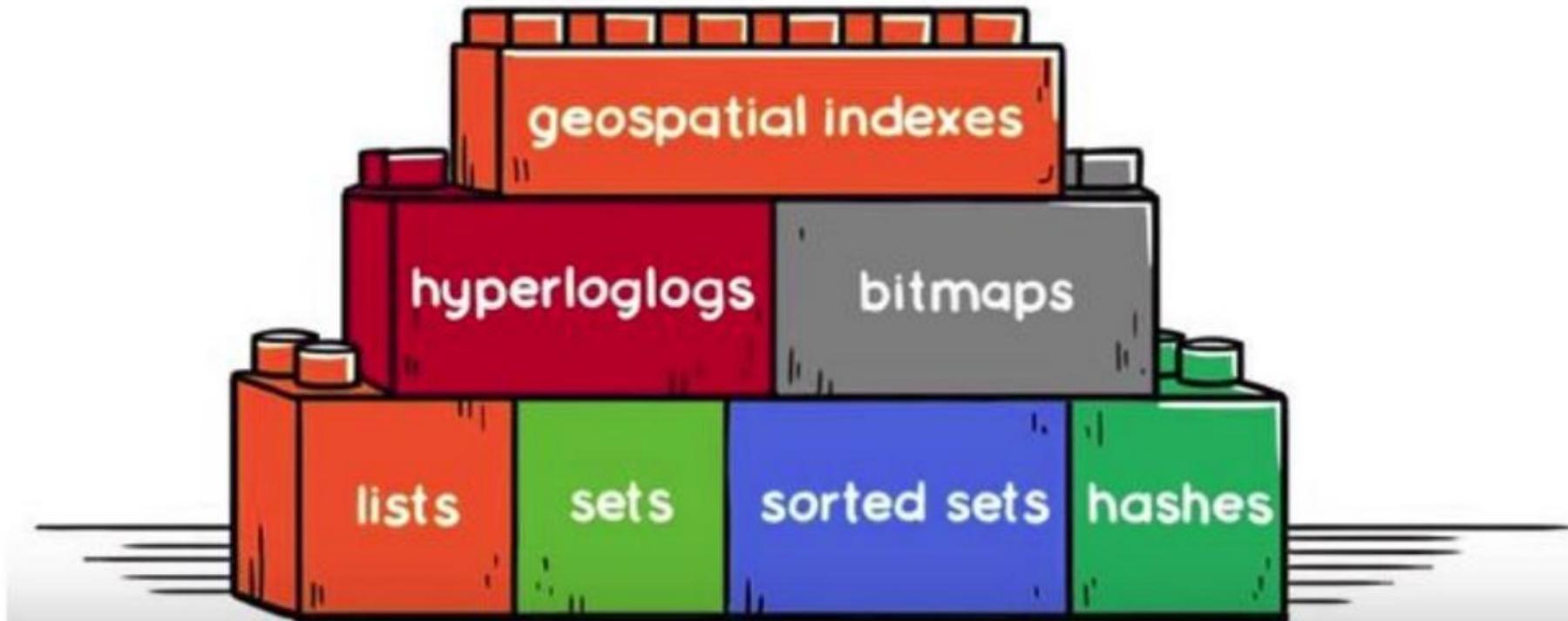


- ❑ FOSS
- ❑ In Memory K/V Store written in C
- ❑ Created for: Caching, Session Store, Queue, Analytics
- ❑ Specific Commands per Data Structures: Strings, Hash, sets
- ❑ Keys TTL
- ❑ Bring your own Data Structure
- ❑ Fast, Low Latency, Battle tested everybody. :D
- ❑ Single Thread :(

Redis data strata

| | | | |
|------|-------------|--------|-----------------|
| v1.0 | Strings | v2.2 | Bit arrays |
| | Lists | v2.8.9 | HyperLogLog |
| | Sets | v3.2 | Geo Sets |
| v1.2 | Sorted Sets | | Bit fields |
| v2.0 | Hashes | v4 | Streams (?) |
| | | v5 | MODULES! |

Redis - Data Structures



Redis >= 4.x

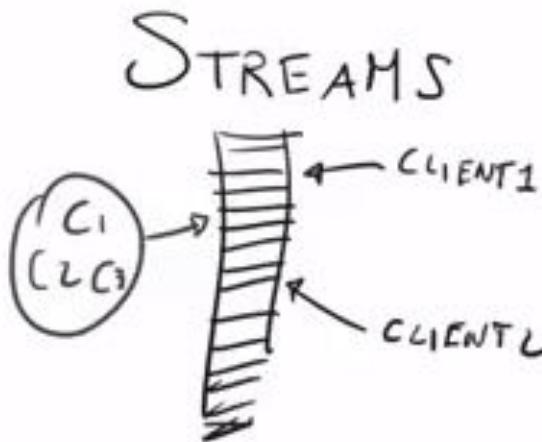


- ❑ Redis-Modules
 - ❑ It's not Lua Scripting
 - ❑ Built in C
 - ❑ Low Latency / Embedded in Redis
 - ❑ Use cases: Extends Redis / New Capabilities DSs

Redis >= 5.x



Big things in 5.0



- TIME SERIES
- IoT
- EVENTS SOURCING
- MESSAGING
- AND MORE ...

Redis: make ; src/redis-server



```
src/redis-server
File Edit View Search Terminal Tabs Help
src/redis-server × src/redis-cli × diego@4winds: ~/github/diegopacheco/sw-design-course/s...
21535:C 22 Nov 2019 14:59:15.198 # o000o000o000o Redis is starting o000o000o000o
21535:C 22 Nov 2019 14:59:15.198 # Redis version=5.0.7, bits=64, commit=00000000, modified=0, pid=21535, just started
21535:C 22 Nov 2019 14:59:15.198 # Warning: no config file specified, using the default config. In order to specify a config file use s
rc/redis-server /path/to/redis.conf

Redis 5.0.7 (00000000/0) 64 bit
Running in standalone mode
Port: 6379
PID: 21535

http://redis.io

21535:M 22 Nov 2019 14:59:15.201 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is s
et to the lower value of 128.
21535:M 22 Nov 2019 14:59:15.201 # Server initialized
21535:M 22 Nov 2019 14:59:15.201 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create laten
cy and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent_hugepage/enabled' as
root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.
21535:M 22 Nov 2019 14:59:15.201 * Ready to accept connections
```

<https://redis.io/download>

Redis: src/redis-cli



```
src/redis-cli
File Edit View Search Terminal Tabs Help
src/redis-server x src/redis-cli x diego@4winds: ~/github/diegopacheco/sw-design-course/s... x 14:59:19 8.83G 1.28
diego@4winds ~> ./bin/redis-5.0.7 src/redis-cli
127.0.0.1:6379> info
# Server
redis_version:5.0.7
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:f34912d9d0bf2646
redis_mode:standalone
os:Linux 4.15.0-70-generic x86_64
arch_bits:64
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:7.4.0
process_id:21535
run_id:4e86b3459c895c04a2cfclc75b26f3214551117f
tcp_port:6379
uptime_in_seconds:17
uptime_in_days:0
hz:10
configured_hz:10
lru_clock:14182484
executable:/home/diego/bin/redis-5.0.7/src/redis-server
config_file:

# Clients
connected_clients:1
client_recent_max_input_buffer:2
client_recent_max_output_buffer:4100800
blocked_clients:0
```

<https://redis.io/download>

Redis: src/redis-cli

```
src/redis-cli
File Edit View Search Terminal Tabs Help
src/redis-server      src/redis-cli      diego@4winds: ~/github/diegopacheco/sw-design-course/s...
127.0.0.1:6379> set counter 1
OK
127.0.0.1:6379> INCR counter
(integer) 2
127.0.0.1:6379> get counter
"2"
127.0.0.1:6379> keys *
1) "counter"
127.0.0.1:6379> HSET person1 name "Diego" email "diego.pachecoit@gmail.com"
(integer) 2
127.0.0.1:6379> HGETALL person1
1) "name"
2) "Diego"
3) "email"
4) "diego.pachecoit@gmail.com"
127.0.0.1:6379> HGET person1 name
"Diego"
127.0.0.1:6379> SADD uniqueStarsSet "Arnold" "Stalone" "The Rock"
(integer) 3
127.0.0.1:6379> SMEMBERS uniqueStarsSet
1) "The Rock"
2) "Stalone"
3) "Arnold"
127.0.0.1:6379> ZADD topPlayers 1 "Ronaldinho" 2 "Ronaldo" 3 "CR" 4 "Romario" 5 "Neymar"
(integer) 5
127.0.0.1:6379> ZRANGE topPlayers 0 2 WITHSCORES
1) "Ronaldinho"
2) "1"
3) "Ronaldo"
4) "2"
5) "CR"
6) "3"
127.0.0.1:6379>
127.0.0.1:6379>
```



<https://redis.io/commands/>

Redis & Java: Lettuce



```
1 import io.lettuce.core.RedisClient;
2 import io.lettuce.core.api.StatefulRedisConnection;
3
4 public class SimpleLettuceMain {
5     Run | Debug
6     public static void main(String[] args) {
7         RedisClient redisClient = RedisClient.create("redis://localhost/0");
8         StatefulRedisConnection<String, String> connection = redisClient.connect();
9
10        System.out.println("Connected to Redis");
11        connection.sync().set("k1", "Hello World");
12        System.out.println("Key from redis k1: " + connection.sync().get("k1"));
13
14        connection.close();
15        redisClient.shutdown();
16    }
17 }
```

```
nov 22, 2019 3:24:23 PM io.lettuce.core.EpollProvider <clinit>
INFO: Starting without optional epoll library
nov 22, 2019 3:24:23 PM io.lettuce.core.KqueueProvider <clinit>
INFO: Starting without optional kqueue library
Connected to Redis
Key from redis k1: Hello World
```

Redis & Java: Spring Data + Lettuce



```
src/redis-cli

File Edit View Search Terminal Tabs Help
src/redis-cli × src/redis-server ×
127.0.0.1:6379> sadd people "diego" "melina" "gandalfy"
(integer) 0
127.0.0.1:6379> SMEMBERS people
1) "melina"
2) "gandalfy"
3) "diego"
127.0.0.1:6379> █
```

Redis & Java: Spring Data + Lettuce



```
1 package people;
2
3 import org.springframework.context.annotation.Bean;
4 import org.springframework.context.annotation.Configuration;
5 import org.springframework.data.redis.connection.RedisStandaloneConfiguration;
6 import org.springframework.data.redis.connection.lettuce.LettuceConnectionFactory;
7 import org.springframework.data.redis.core.RedisTemplate;
8
9 @Configuration
10 class AppConfig {
11
12     @Bean
13     public LettuceConnectionFactory redisConnectionFactory() {
14         return new LettuceConnectionFactory(new RedisStandaloneConfiguration("127.0.0.1", 6379));
15     }
16
17     @Bean
18     public RedisTemplate<?, ?> redisTemplate() {
19         RedisTemplate<byte[], byte[]> template = new RedisTemplate<>();
20         template.setConnectionFactory(redisConnectionFactory());
21         return template;
22     }
23
24 }
25 }
```

Redis & Java: Spring Data + Lettuce



src/main/java/people/SimpleService.java (1 people)

```
1 package people;
2
3 import java.util.Set;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.data.redis.core.RedisTemplate;
7 import org.springframework.stereotype.Service;
8
9 @Service
10 public class SimpleService {
11
12     @Autowired
13     RedisTemplate<String, String> redis;
14
15     public Set<String> getAllPeople(String key) {
16         return redis.opsForSet().members(key);
17     }
18
19 }
20
```

Redis & Java: Spring Data + Lettuce



```
1 package people;
2
3 import org.springframework.boot.CommandLineRunner;
4 import org.springframework.boot.SpringApplication;
5 import org.springframework.boot.autoconfigure.SpringBootApplication;
6 import org.springframework.context.ApplicationContext;
7 import org.springframework.context.annotation.Bean;
8
9 @SpringBootApplication
10 public class SpringDataRedisMain {
11
12     Run | Debug
13     public static void main(String[] args) {
14         SpringApplication.run(SpringDataRedisMain.class, args);
15     }
16
17     @Bean
18     public CommandLineRunner commandLineRunner(ApplicationContext ctx) {
19         return args -> {
20
21             System.out.println("Running Spring Boot application in Console... ");
22
23             SimpleService service = ctx.getBean(SimpleService.class);
24             System.out.println("All People :" + service.getAllPeople("people"));
25
26         };
27     }
28 }
```

Redis & Java: Spring Data + Lettuce

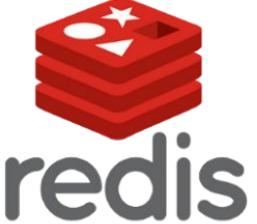


Redis & Java: Spring Data + Lettuce : Hash Mapping



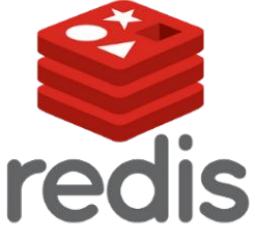
```
1 package people;
2
3 import org.springframework.data.annotation.Id;
4 import org.springframework.data.redis.core.RedisHash;
5
6 @RedisHash("players")
7 public class Person{
8     @Id String id;
9     String name;
10
11     public Person(){}
12
13     public Person(String id,String name){
14         this.id=id;
15         this.name=name;
16     }
17
18     public String getId() {
19         return id;
20     }
21     public void setId(String id) {
22         this.id = id;
23     }
24     public String getName() {
25         return name;
26     }
27     public void setName(String name) {
28         this.name = name;
29     }
30     @Override
31     public String toString() {
32         return "id: " + id + " name: " + name;
```

Redis & Java: Spring Data + Lettuce : Hash Mapping



```
1 package people;  
2  
3 import org.springframework.data.repository.CrudRepository;  
4  
5 public interface PersonRepository extends CrudRepository<Person, String> {}  
6
```

Redis & Java: Spring Data + Lettuce : Hash Mapping



```
3 import org.springframework.boot.CommandLineRunner;
4 import org.springframework.boot.SpringApplication;
5 import org.springframework.boot.autoconfigure.SpringBootApplication;
6 import org.springframework.context.ApplicationContext;
7 import org.springframework.context.annotation.Bean;
8
9 @SpringBootApplication
10 public class SpringDataRedisRepositoryMain{
11
12     Run | Debug
13     public static void main(String[] args) {
14         SpringApplication.run(SpringDataRedisRepositoryMain.class, args);
15     }
16
17     @Bean
18     public CommandLineRunner commandLineRunnerMapper(ApplicationContext ctx) {
19         return args -> {
20
21             System.out.println("Running Spring Boot application in Console... ");
22
23             PersonRepository repo = ctx.getBean(PersonRepository.class);
24             Person p = new Person("diegopacheco", "diego.pacheco.it@gmail.com");
25             repo.save(p);
26             System.out.println("Persons: " + repo.findAll());
27             System.out.println("Count: " + repo.count());
28             repo.delete(p);
29         };
30     }
31 }
```

Redis & Java: Spring Data + Lettuce : Hash Mapping



Exercises



Constraints

You can code this exercises with Java, Scala or Clojure.

You can use any framework you like. UI is not required.

You need to run Cassandra 3.x and Redis 5.x (natively, docker, does not matter).

1. Use cassandra and build Movie Catalog Service Where movies should have ID(UUID), name, category, length, linkToYouTubeTrailer. You should be able to query by category, name and ID.
2. Use Redis and Build a Twitter application where you should have tweet(message) and you should be able to follow people and people could follow you.
3. Build a profile Service using Cassandra, you should have a REST interface with the following data: ID(UUID), name, email, dateOfBirth, gender, twitterURL, faceURL. You need store the data in Cassandra and use Redis as a Cache. You need support the following operations:
 - a. List all users by Id
 - b. List all users by Email
 - c. Add / Remove / Update users



Cassandra & Redis

DIEGO PACHECO