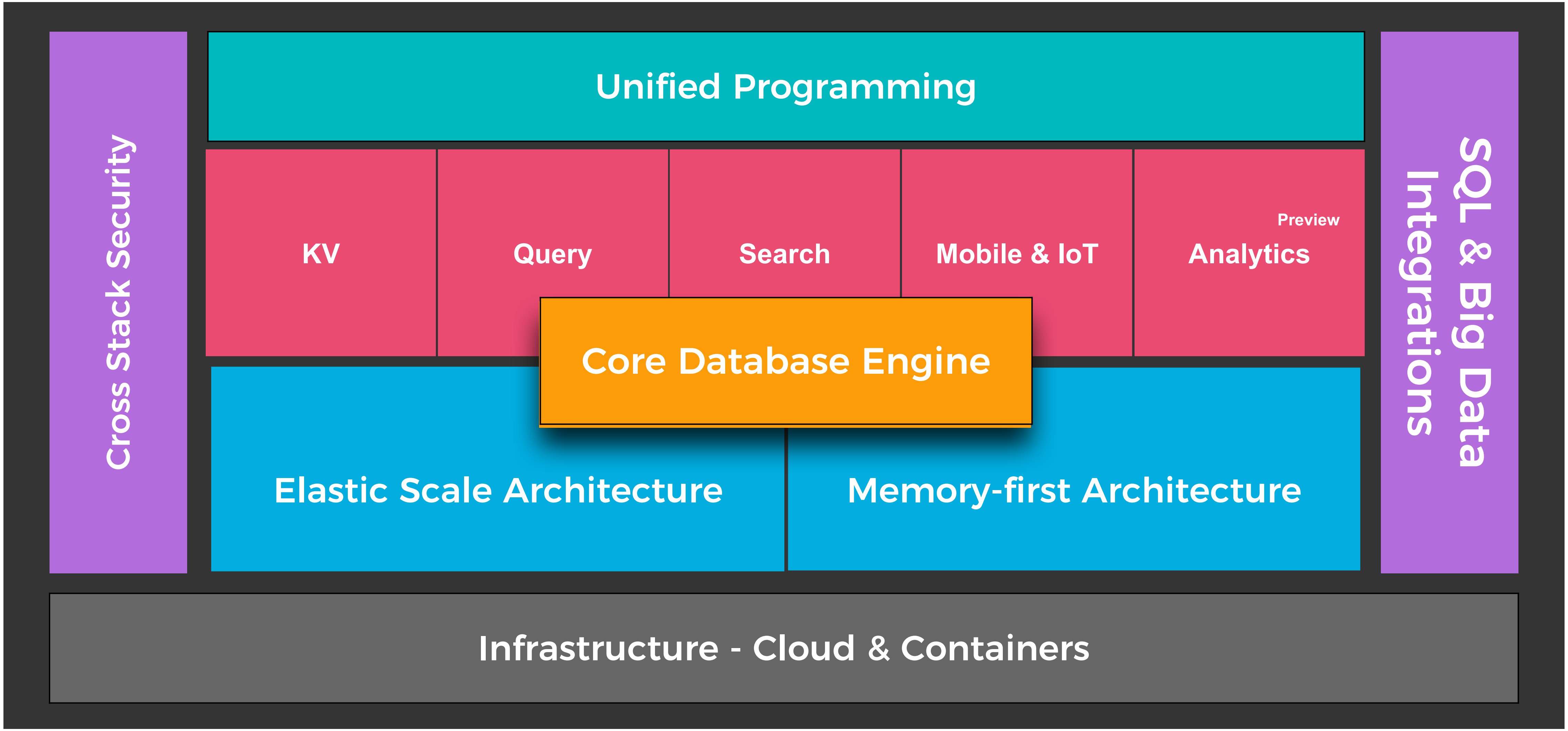
Couchbase

Architecture and Administration Basics

Workshop Day 1 - Architecture

Confidential and Proprietary. Do not distribute without Couchbase consent. © Couchbase 2017. All rights reserved.

Couchbase Data Platform



Unified Programming

KV

Query

Search

Mobile & IoT

Preview

Analytics

Core Database Engine

Elastic Scale Architecture

Memory-first Architecture

Cross Stack Security

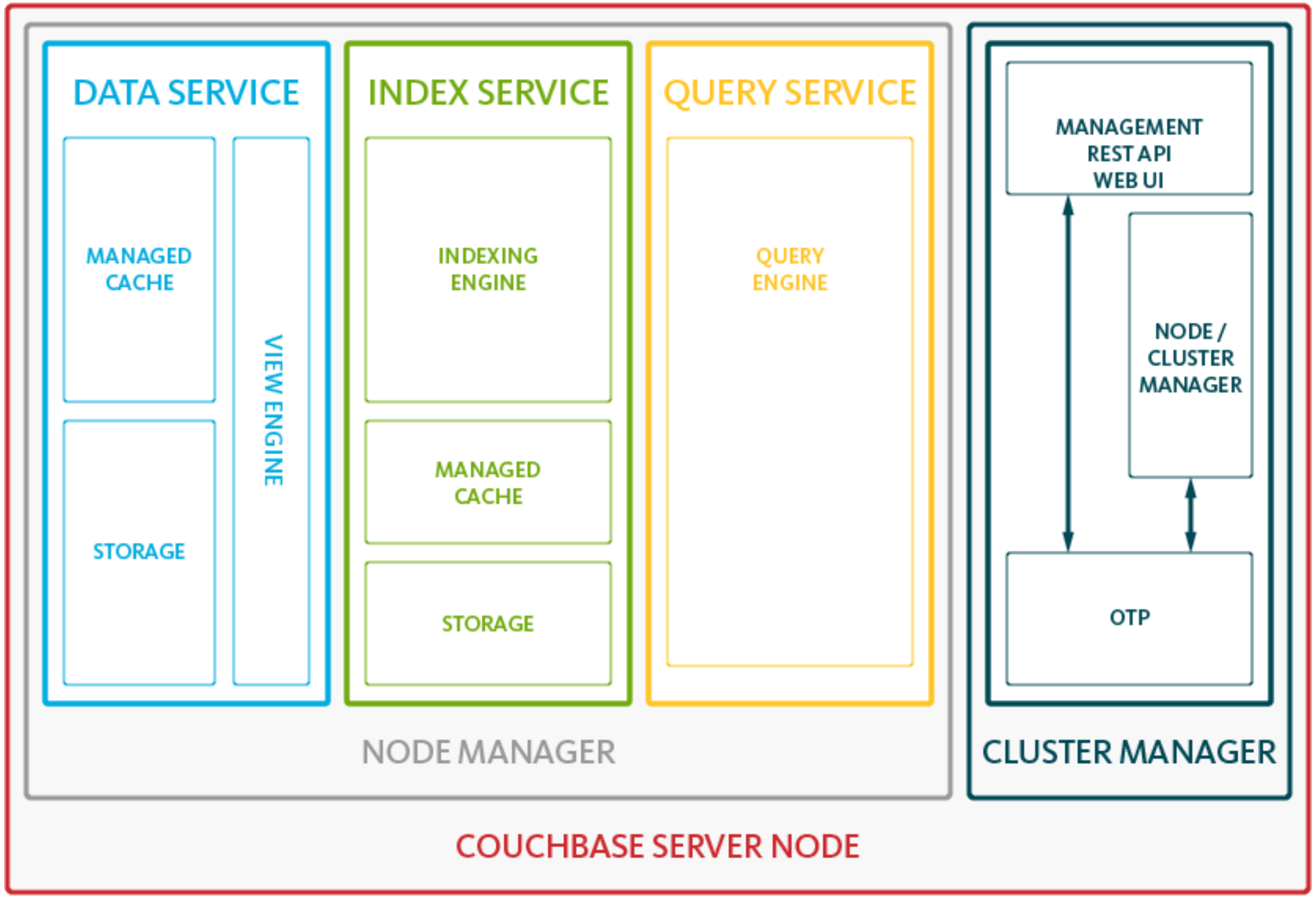
SQL & Big Data Integrations

Infrastructure - Cloud & Containers

1 | Architecture

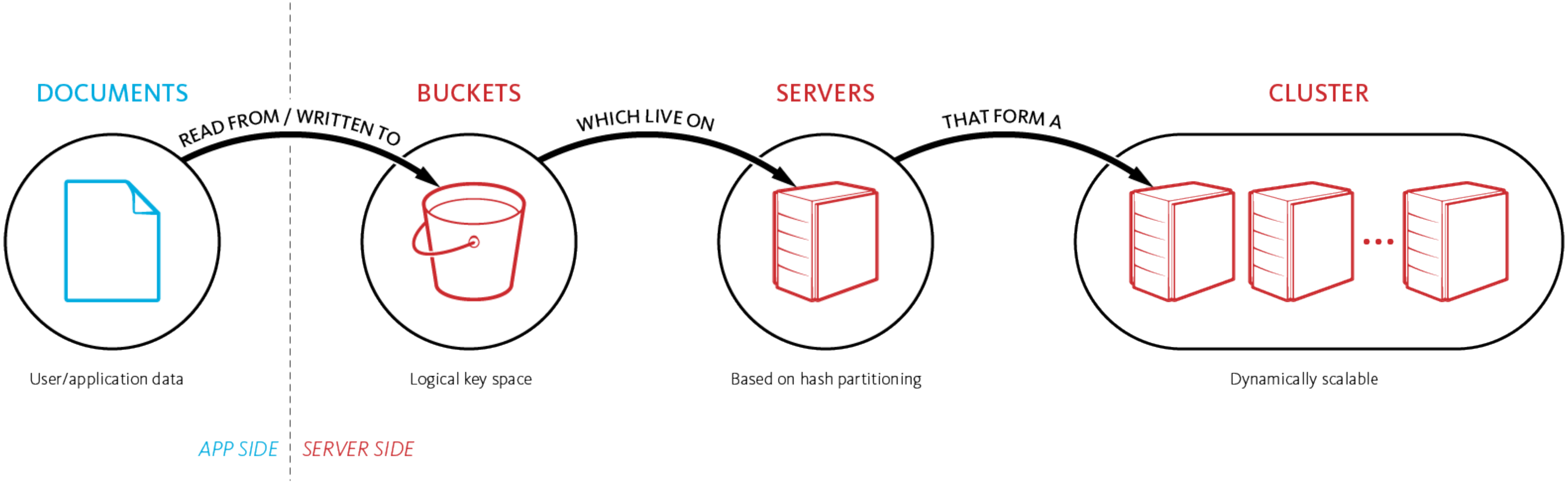
Confidential and Proprietary. Do not distribute without Couchbase consent. © Couchbase 2017. All rights reserved.

Couchbase Architecture



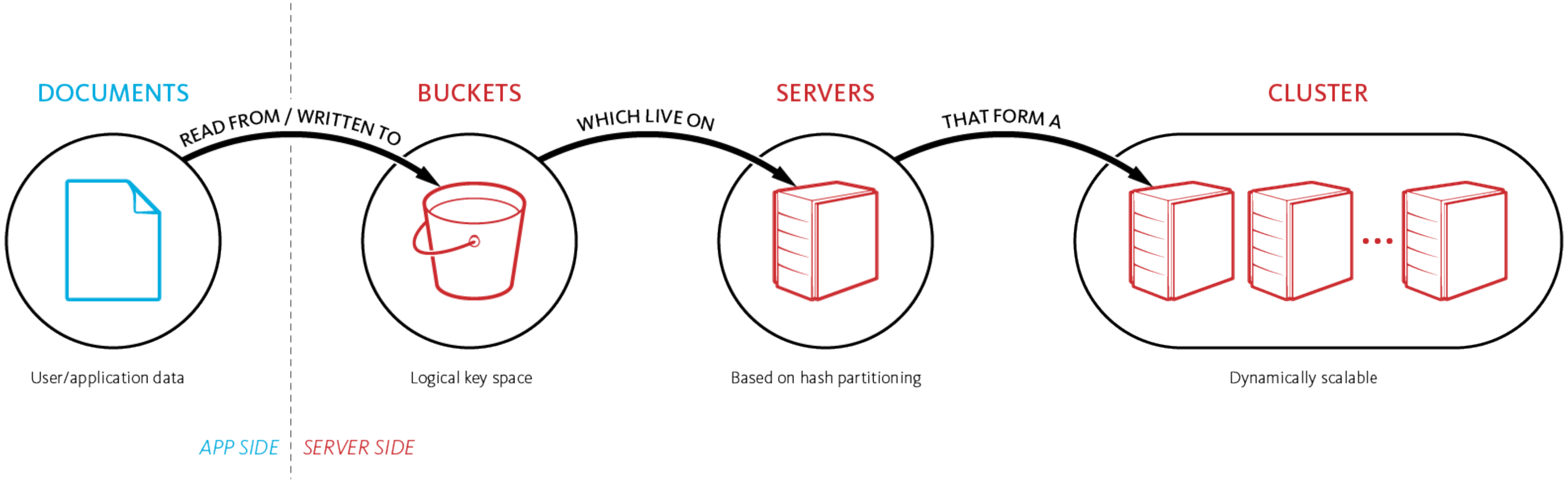
- Data Service – Key Value Store and builds and maintains Distributed secondary indexes (MapReduce Views)
- Indexing Engine – builds and maintains Global Secondary Indexes
- Query Engine – plans, coordinates, and executes queries against either Global or Distributed indexes
- Cluster Manager – configuration, heartbeat, statistics, RESTful Management interface

Storing And Retrieving Documents

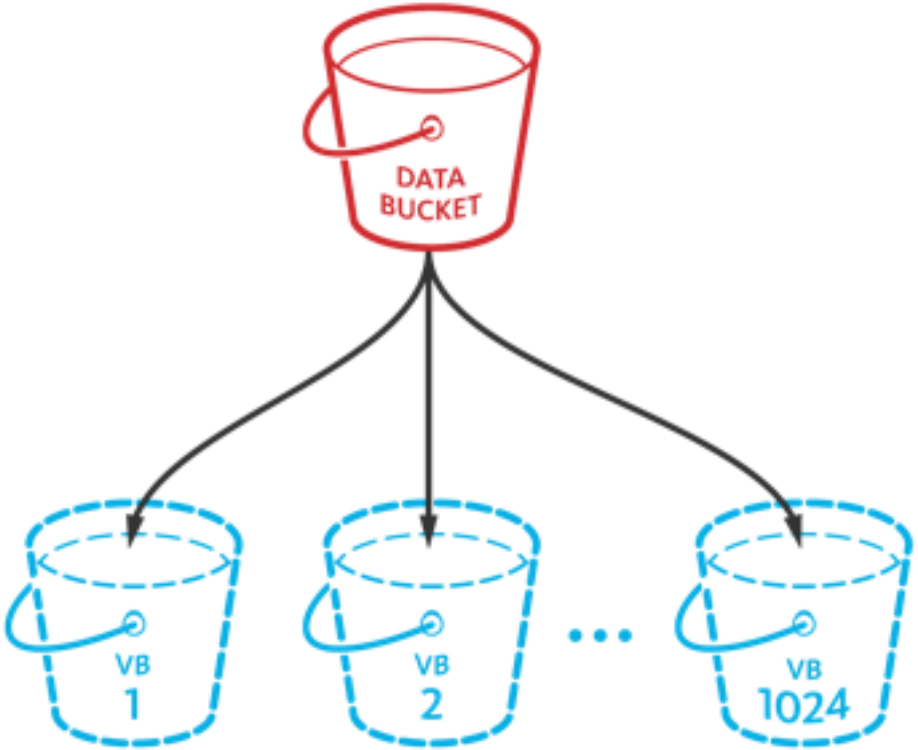


Buckets - When to use more than one ?

- When you need to treat or access the data differently
  - Different High Availability requirements (1,2 or 3 replicas)
  - Different performance / residency needs (how much data to cache)
  - Security / Multi-tenancy
  - Segregating Binary and JSON data – especially with view usage



Auto sharding – Bucket and vBuckets



Virtual buckets

- Bucket
  - A bucket is a logical, unique key space
  - Multiple buckets can exist within a single cluster of nodes
- vBuckets
  - Each bucket has active and replica data sets (1, 2 or 3 **extra** copies)
  - Each data set has 1024 Virtual Buckets (vBuckets)
  - Each vBucket contains 1/1024th portion of the data set
  - vBuckets do not have a fixed physical server location
  - Mapping between the vBuckets and physical servers is called the cluster map
  - Document IDs (keys) always get hashed to the same vBucket (consistent hashing)
  - Couchbase SDK's lookup the vBucket -> server mapping

Bucket Comparison

	Memcached	Couchbase	Ephemeral New in 5.0
Persistence	✗	✓	✗
Replication	✗	✓	✓
Rebalance	✗	✓	✓
XDCR	✗	✓	✓
N1QL	✗	✓	✓
Indexing	✗	✓	✓*
Max Object Size	1MB	20MB	20MB

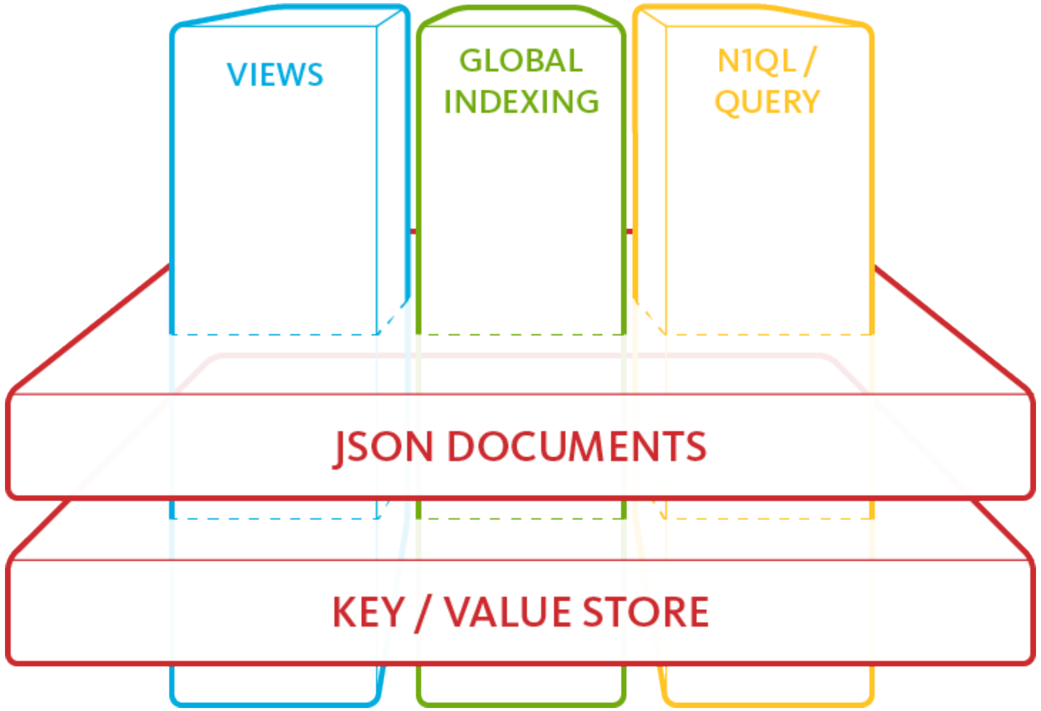
* MOI, FTS only

Ephemeral Bucket Benefits and Limitations

- Benefits
  - No high performance disk subsystem required
    - Lower cost VMs
    - Smaller chassis
  - Even more consistent high performance
    - No disk IO contention (i.e. compaction)
  - Lower CPU consumption
    - No Disk Write Queue
    - No IO threads
  - Faster maintenance operations
    - No warm-up
    - Faster node restart
    - Faster rebalance – currently 4x faster in our lab!

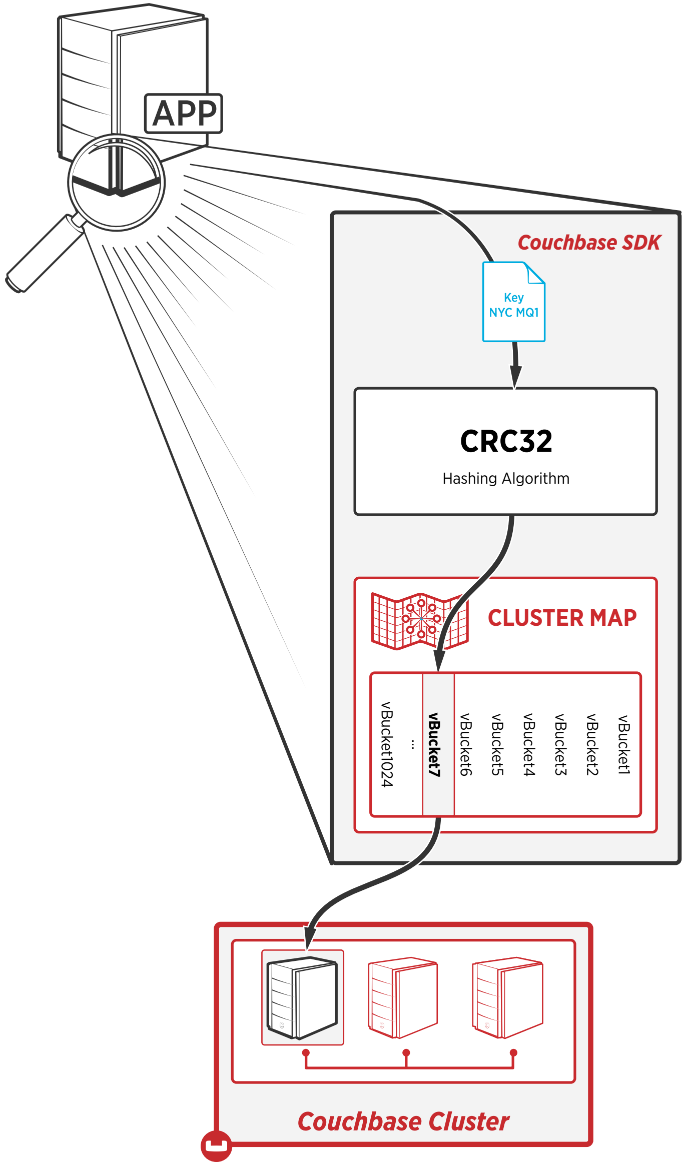
- Limitations
  - Data set must fit in memory
    - Configurable OOM handling
  - No automatic recovery from total power loss
    - Backups and XDCR still supported!
  - XDCR limitations
    - Ephemeral to Ephemeral
    - Couchbase to Ephemeral
    - Ephemeral to Couchbase NOT supported
  - Only Memory Optimized Indexes (MOI) and Full Text Search (FTS) are supported
    - No Views or Standard GSI

Couchbase Data Access

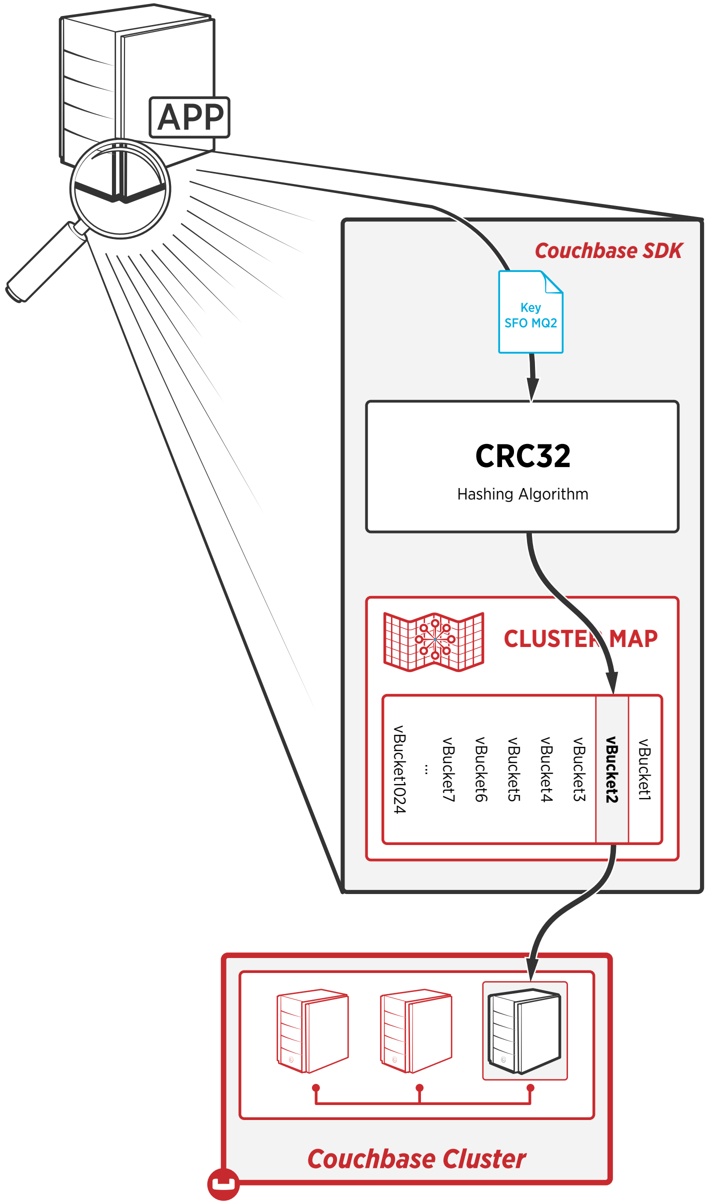


- Everything is layered on top of Key Value
- A Document store is a special case of Key-Value
- Views provide aggregation and real-time analytics through incremental map-reduce
- Global Secondary Indexes provide low latency/high throughput indexes
- N1QL is a language that provides a powerful and expressive way of accessing documents

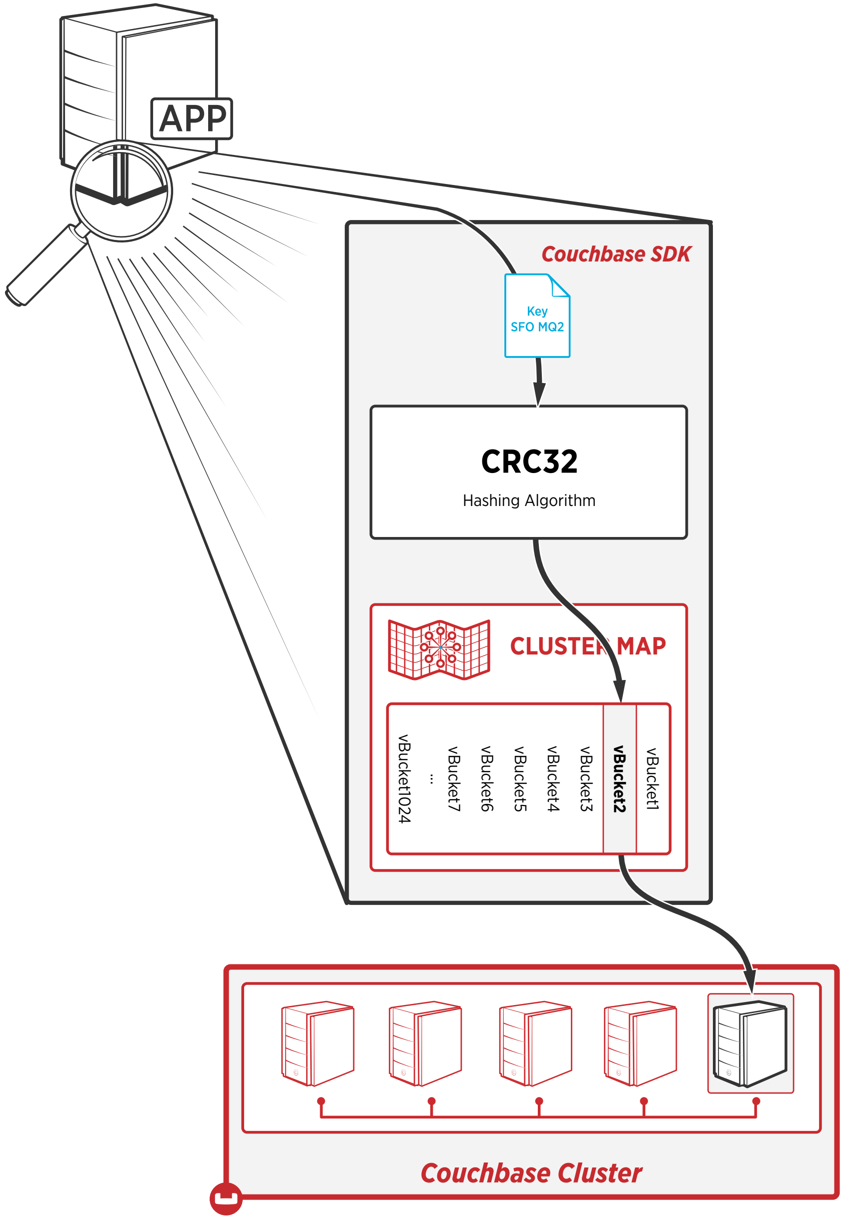
Cluster Map


APP
Couchbase SDK
Key
NYC MQ1
CRC32
Hashing Algorithm
CLUSTER MAP
vBucket1
vBucket2
vBucket3
vBucket4
vBucket5
vBucket6
vBucket7
...
vBucket1024
Couchbase Cluster

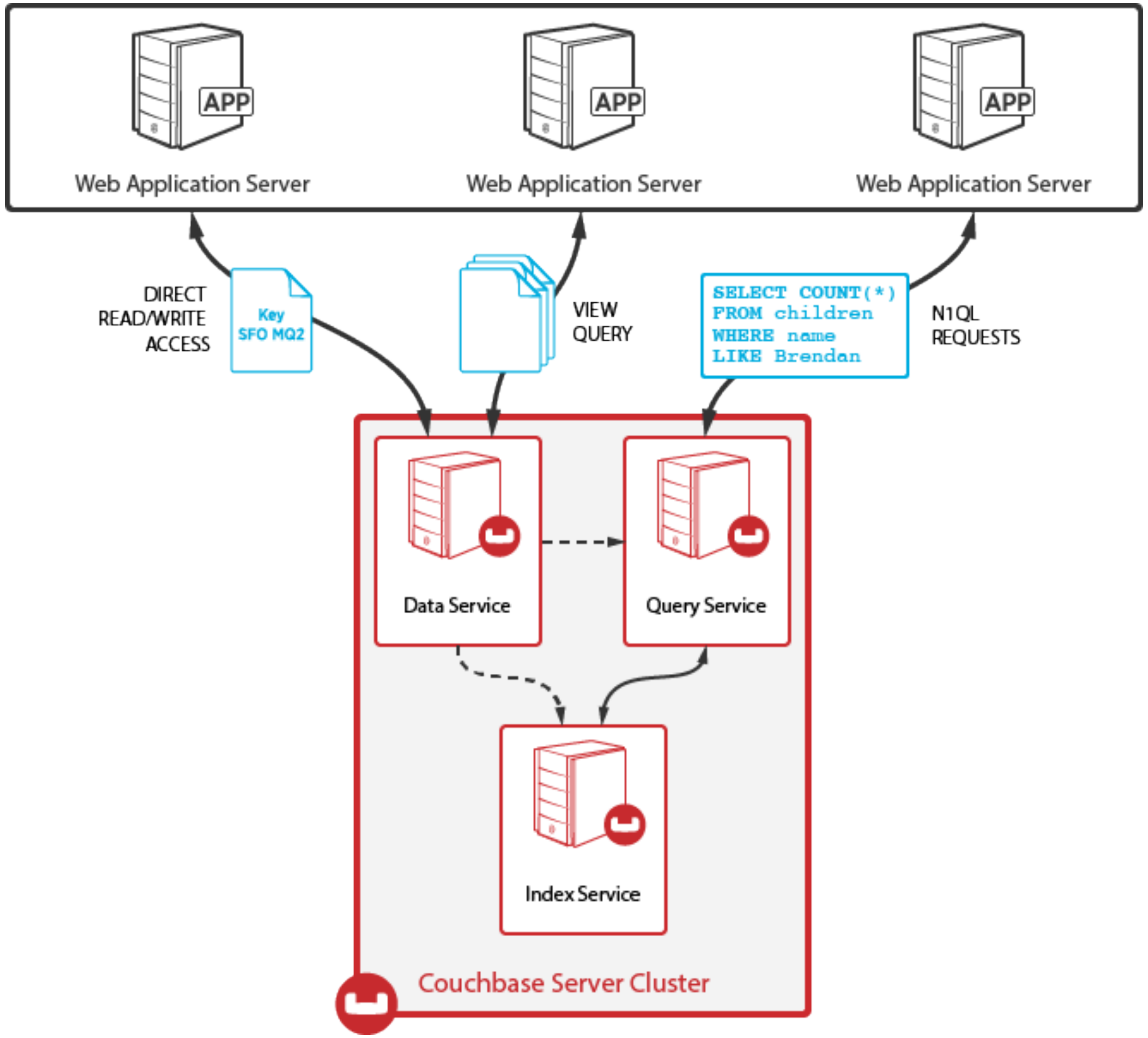
Cluster Map


APP
Couchbase SDK
Key
SFO MQ2
CRC32
Hashing Algorithm
CLUSTER MAP
vBucket1
vBucket2
vBucket3
vBucket4
vBucket5
vBucket6
vBucket7
...
vBucket1024
Couchbase Cluster

Cluster Map – Addition of 2 Nodes



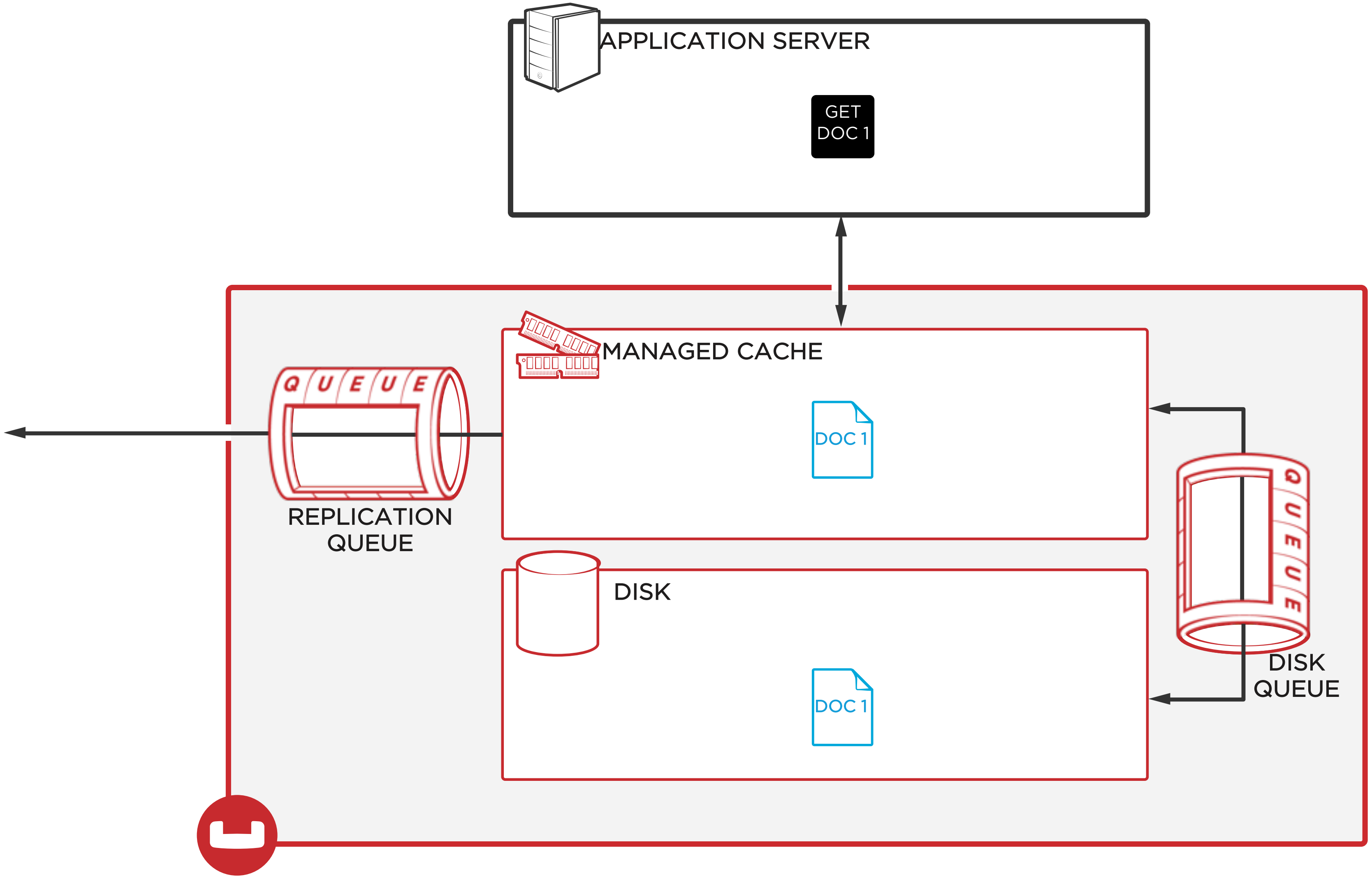
Application to Database Interaction



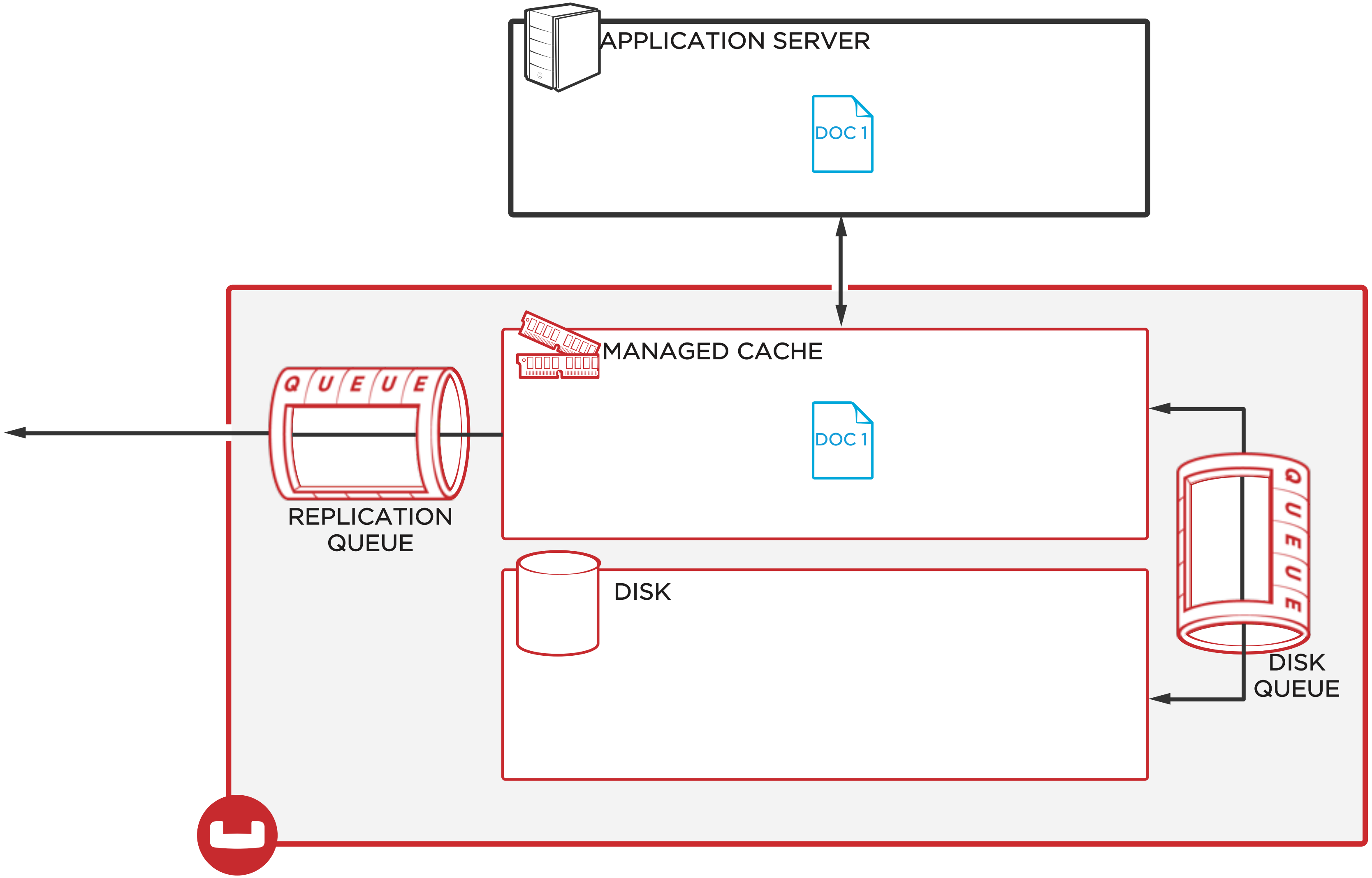
```
SELECT COUNT(*)
FROM children
WHERE name
LIKE Brendan
```

2 SDK Operations

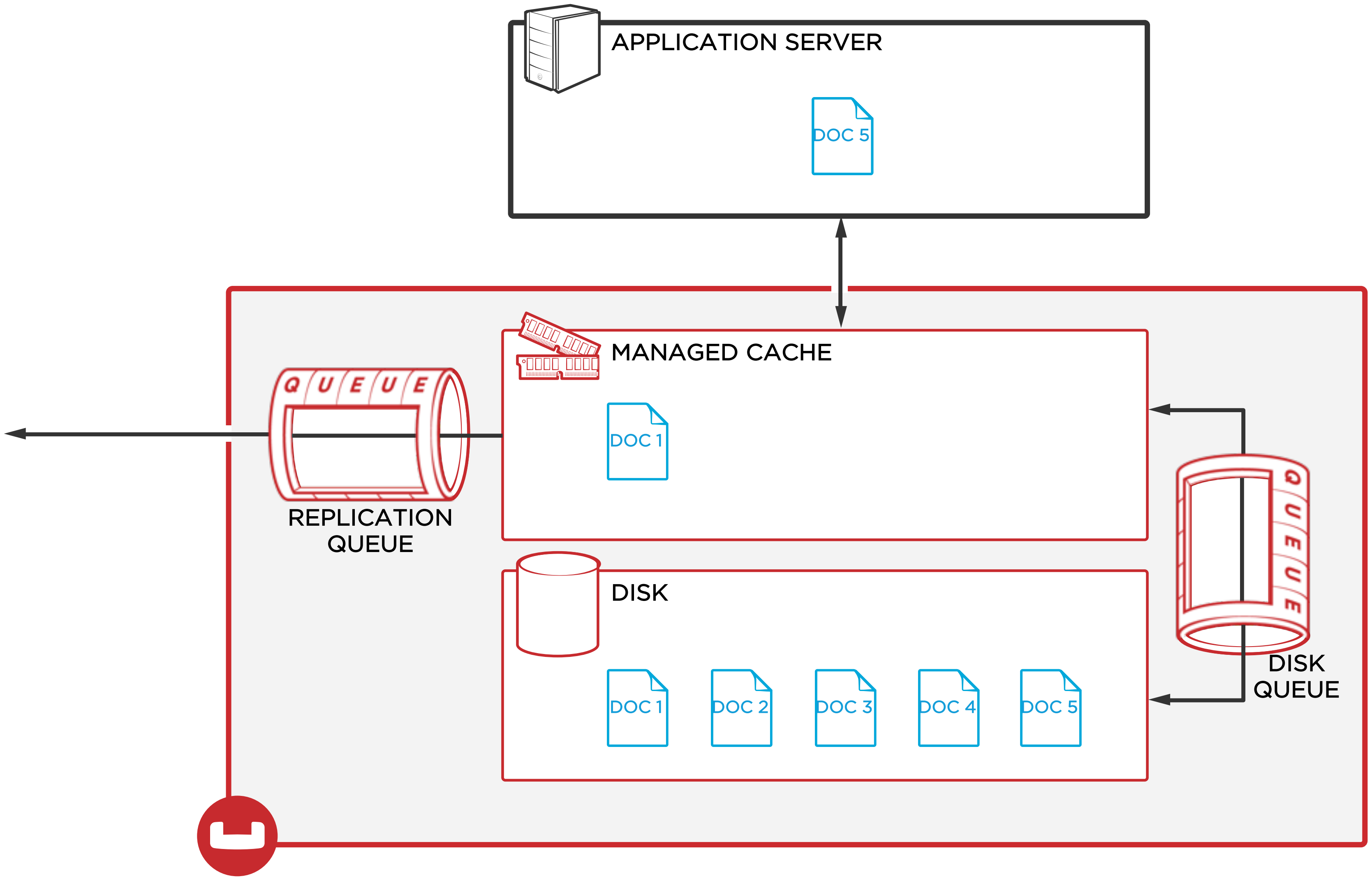
Couchbase Read Operation



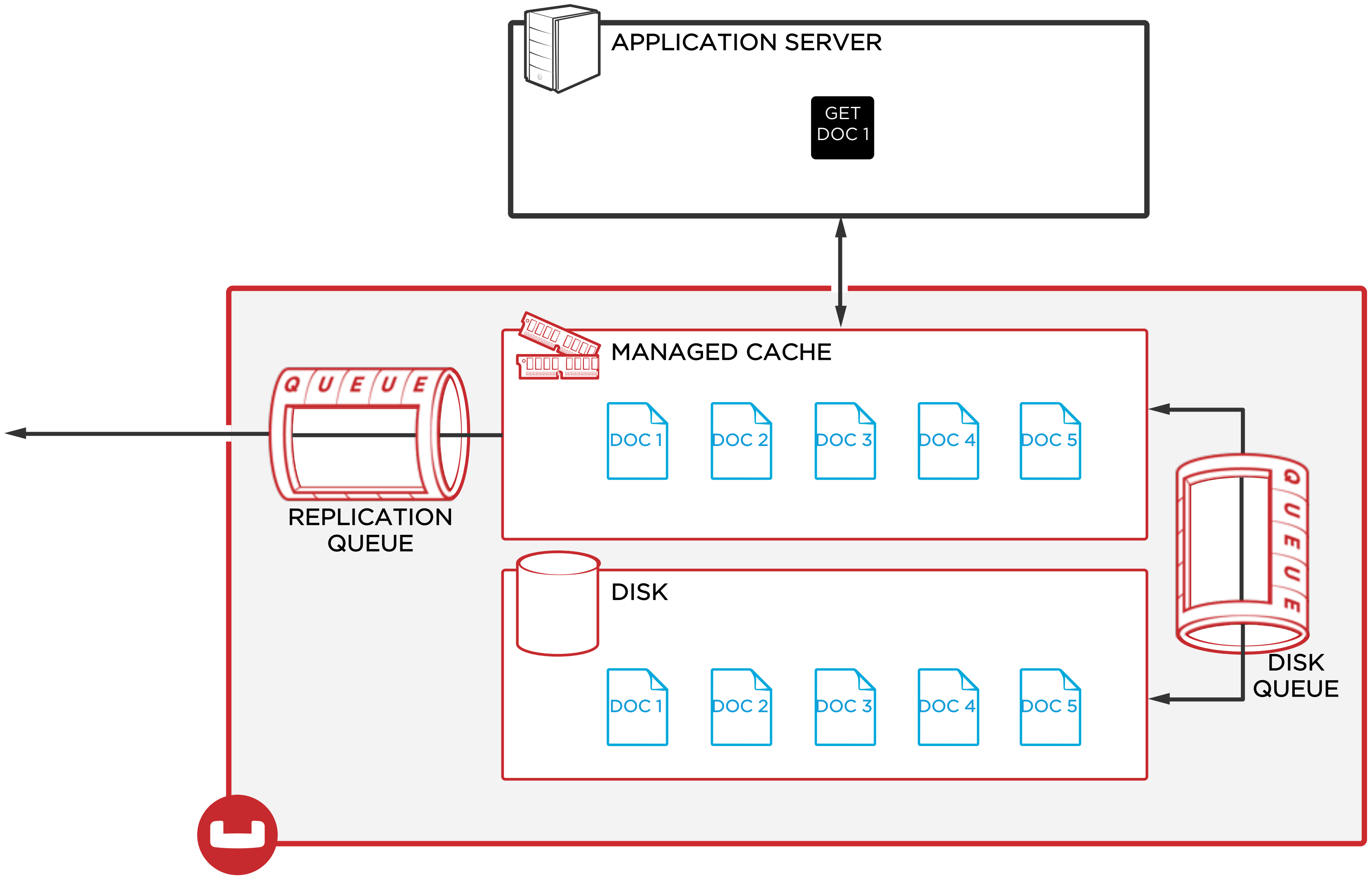
Write Operation



Cache Ejection



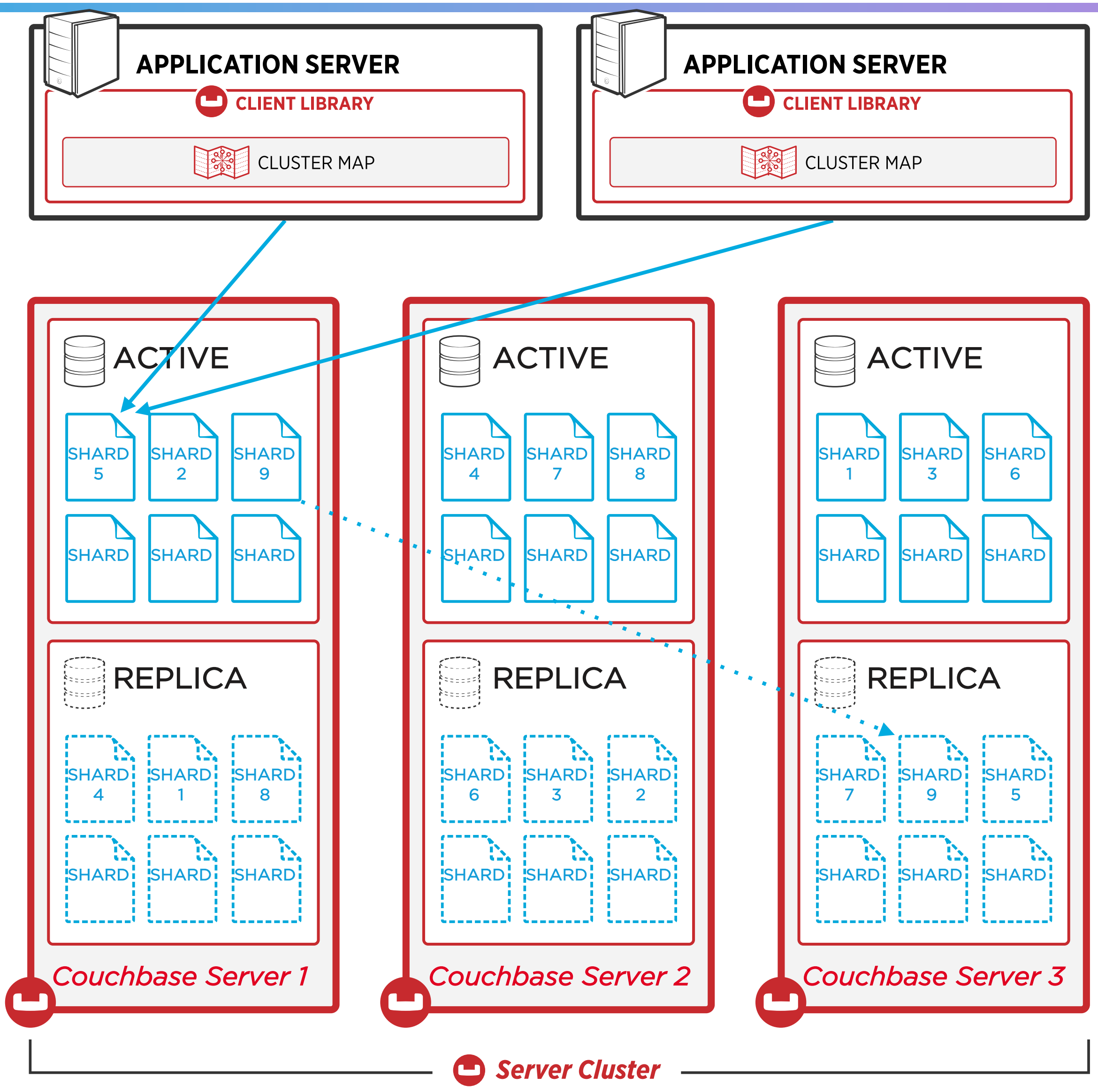
Cache Miss


APPLICATION SERVER
GET DOC 1
MANAGED CACHE
DOC 1
DOC 2
DOC 3
DOC 4
DOC 5
QUEUE
REPLICATION QUEUE
DISK
DOC 1
DOC 2
DOC 3
DOC 4
DOC 5
QUEUE
DISK QUEUE

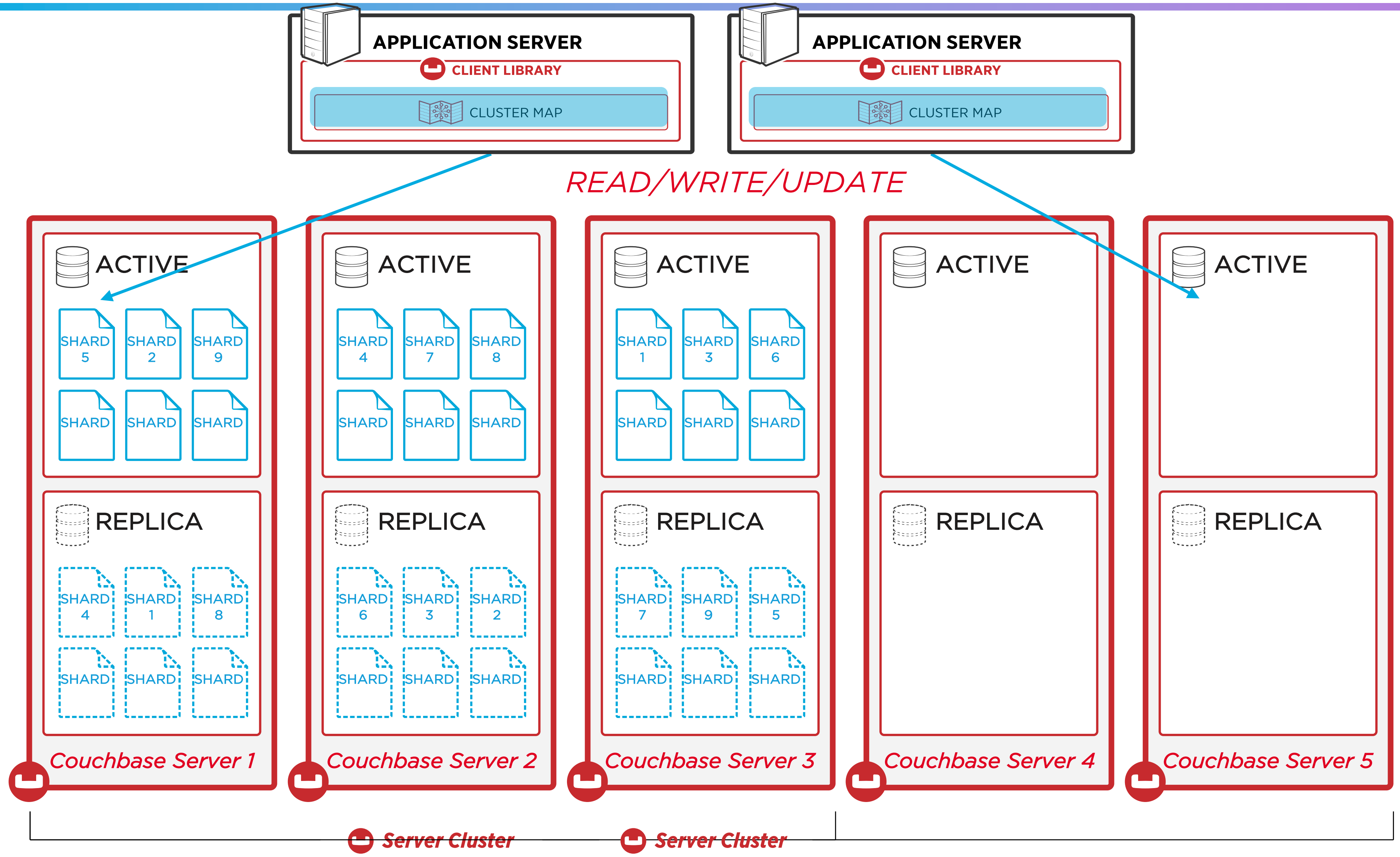
3 Cluster Operations

Confidential and Proprietary. Do not distribute without Couchbase consent. © Couchbase 2017. All rights reserved.

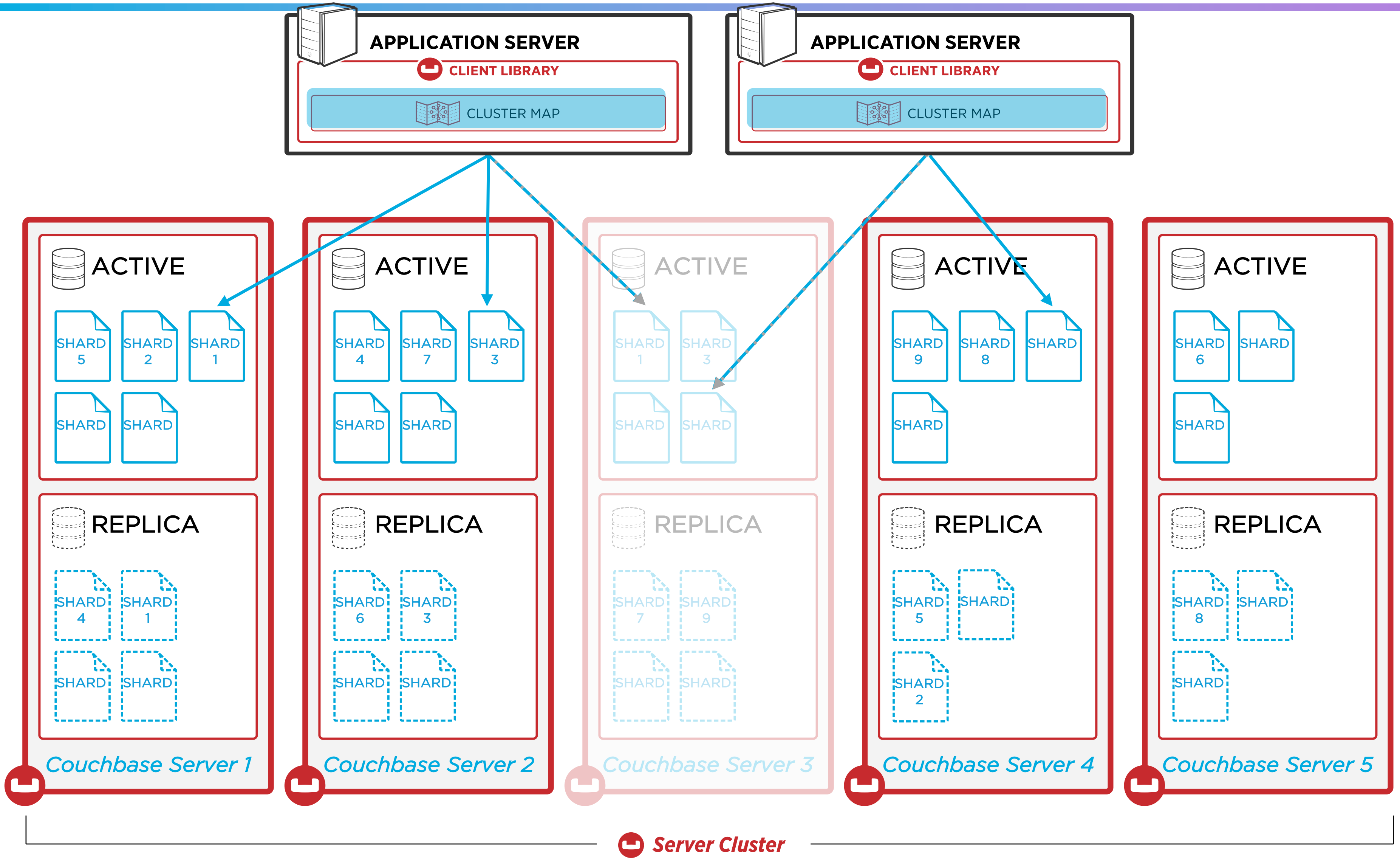
Basic Operation



Add Nodes to Cluster



Fail Over Node

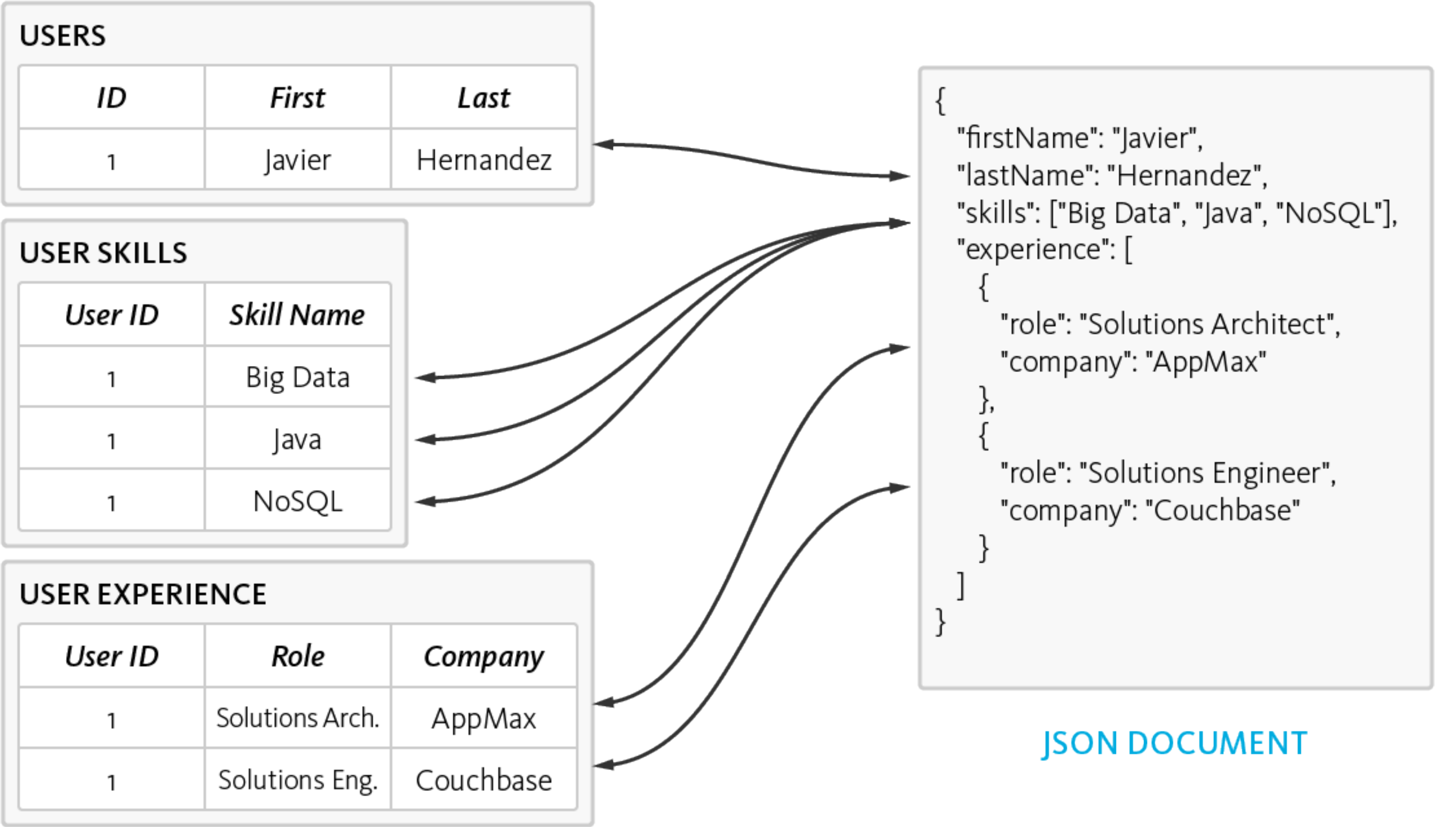


JSON Support

Flexibility

The Power of the Flexible JSON Schema

- Ability to store data in multiple ways
  - Denormalized single document, as opposed to normalizing data across multiple table
  - Dynamic Schema to add new values when needed



USERS

ID	First	Last
1	Javier	Hernandez

USER SKILLS

User ID	Skill Name
1	Big Data
1	Java
1	NoSQL

USER EXPERIENCE

User ID	Role	Company
1	Solutions Arch.	AppMax
1	Solutions Eng.	Couchbase

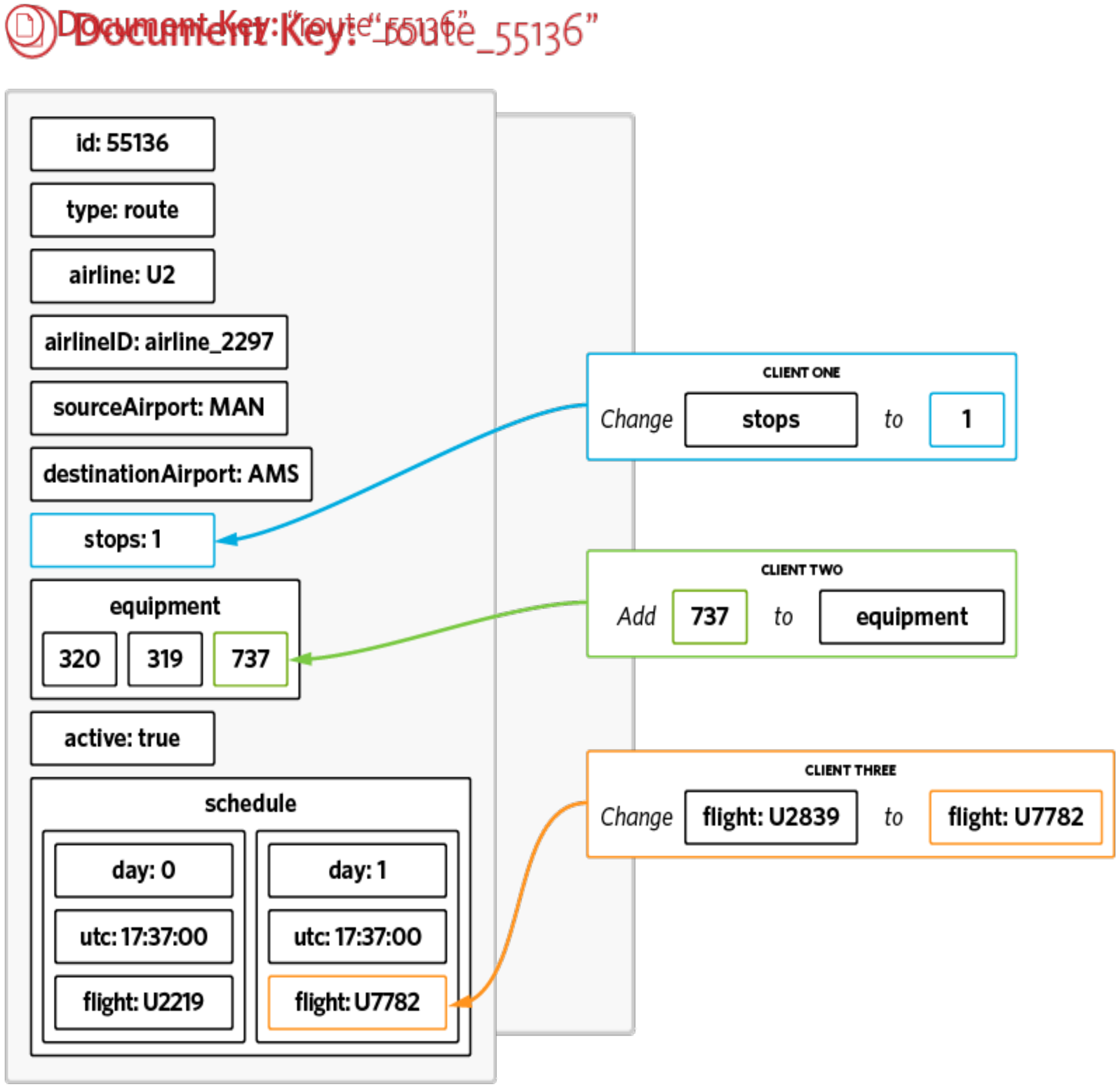
RELATIONAL TABLES

```
{
  "firstName": "Javier",
  "lastName": "Hernandez",
  "skills": ["Big Data", "Java", "NoSQL"],
  "experience": [
    {
      "role": "Solutions Architect",
      "company": "AppMax"
    },
    {
      "role": "Solutions Engineer",
      "company": "Couchbase"
    }
  ]
}
```

JSON DOCUMENT

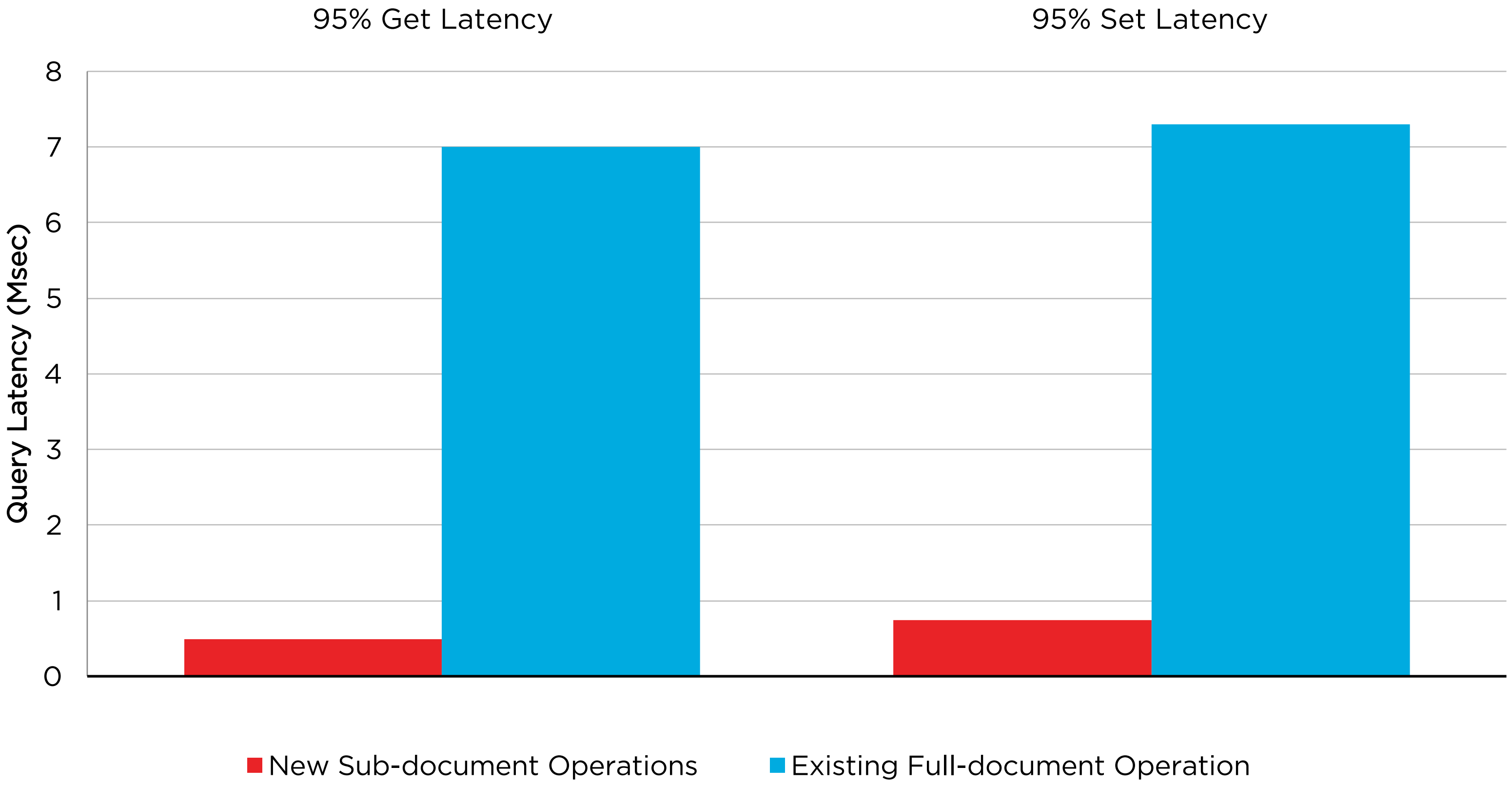
Efficient Sub-Document Operations

- Document Mutations:
  - Atomic Operate on individual fields
  - Identical syntax behavior to regular bucket methods (upsert, insert, get, replace)
  - Support for JSON fragments.
  - Support for Arrays with uniqueness guarantees and ordinal placement (front/back)



Faster Key Based Operations!

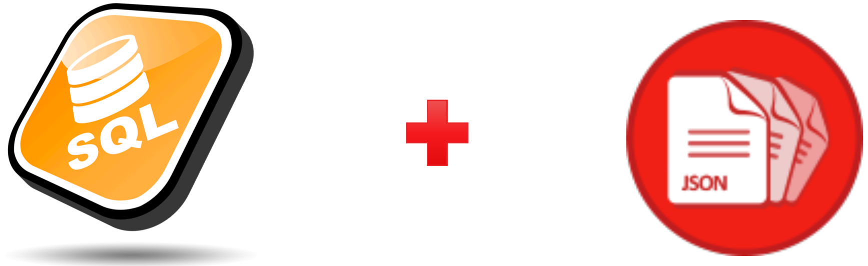
10-14X Faster Document Read & Update Operation



95% Get & Set Latency Measured (Msecs) on single INT field update with sub-document vs full-document call

Nickel (N1QL) : SQL-Like Querying Support

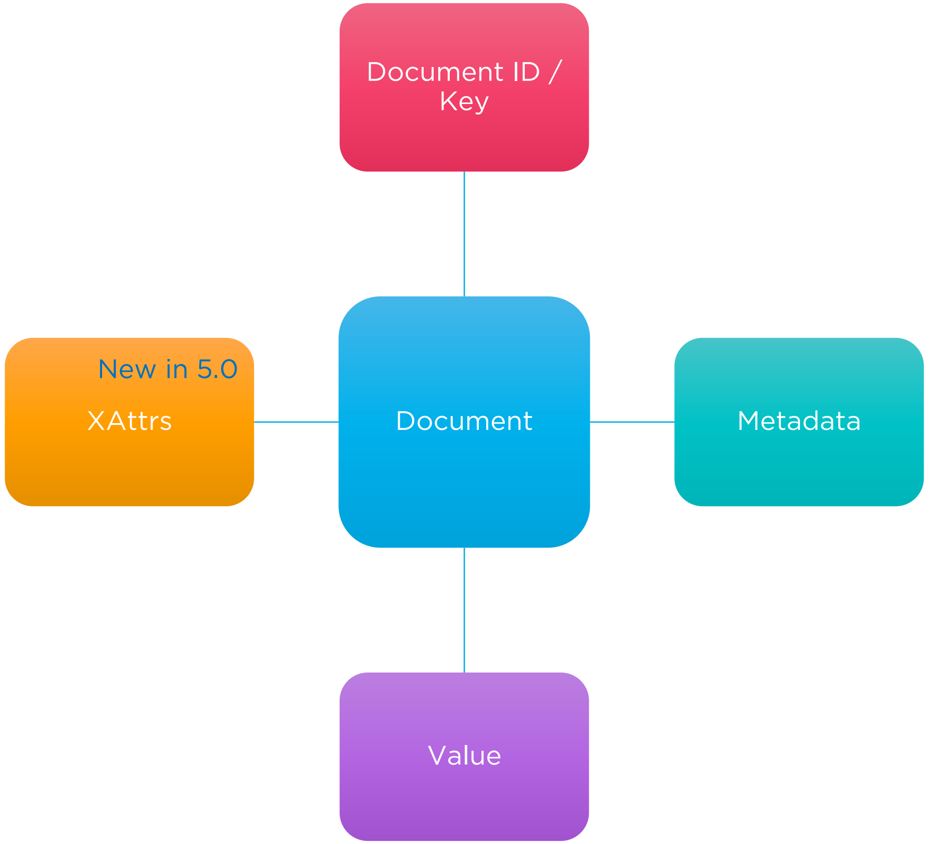
- SQL-like Query Language
  - Expressive, familiar, and feature-rich language for querying, transforming, and manipulating JSON data
  - ANSI 92 SQL Compatible – Selects, Inserts, Updates, Group By, Sort, Functions etc.
- N1QL extends SQL to handle data that is:
  - **Nested**: Contains nested objects, arrays
  - **Heterogeneous**: Schema-optional, non-uniform
  - **Distributed**: Partitioned across a cluster



Power of SQL

Flexibility of JSON

JSON Document Support

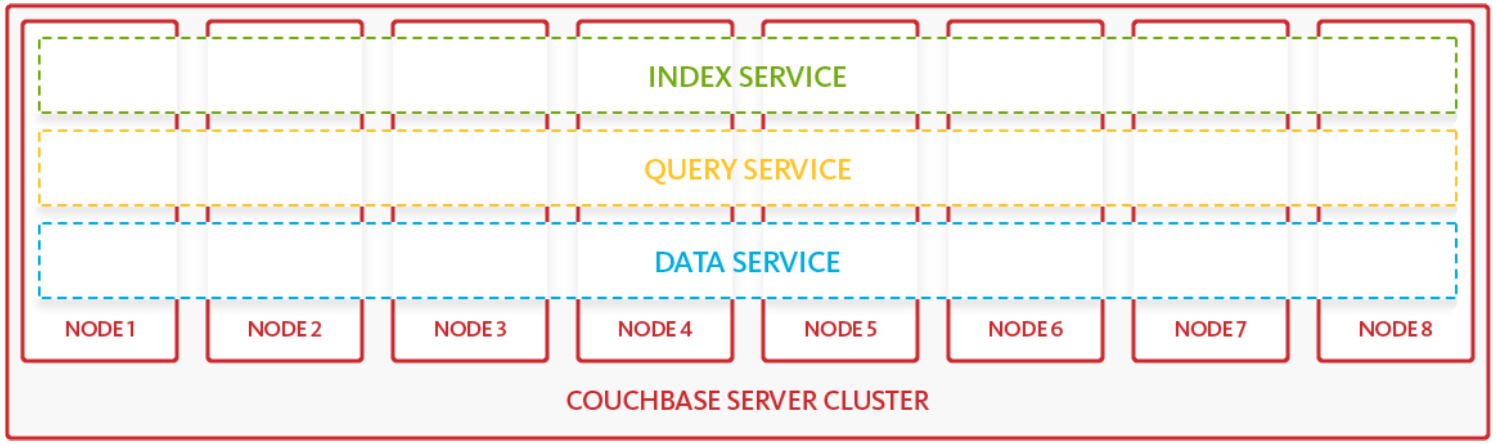


- Document ID / Key (Max 250 bytes):
  - Must be unique / Lookup is extremely fast
  - Similar to primary keys in relational databases
  - Documents are partitioned based on the document ID
- Value (Max 20 MB)
  - JSON
  - Binary - integers, strings, booleans
  - Common binary values include serialized objects, compressed XML, compressed text, encrypted values
- Metadata (Fixed 56 bytes)
  - CAS Value (unique identifier for concurrency)
  - TTL
  - Flags (optional client library metadata)
  - Revision ID #
- XAttr (Max 20 MB) New in 5.0
  - Non-enumerable eXtended Attributes

4 MDS

Modern Architecture – Multi-Dimensional Scaling

MDS is the architecture that enables independent scaling of data, query, and indexing workloads while being managed as one cluster.



Modern Architecture – Multi-Dimensional Scaling

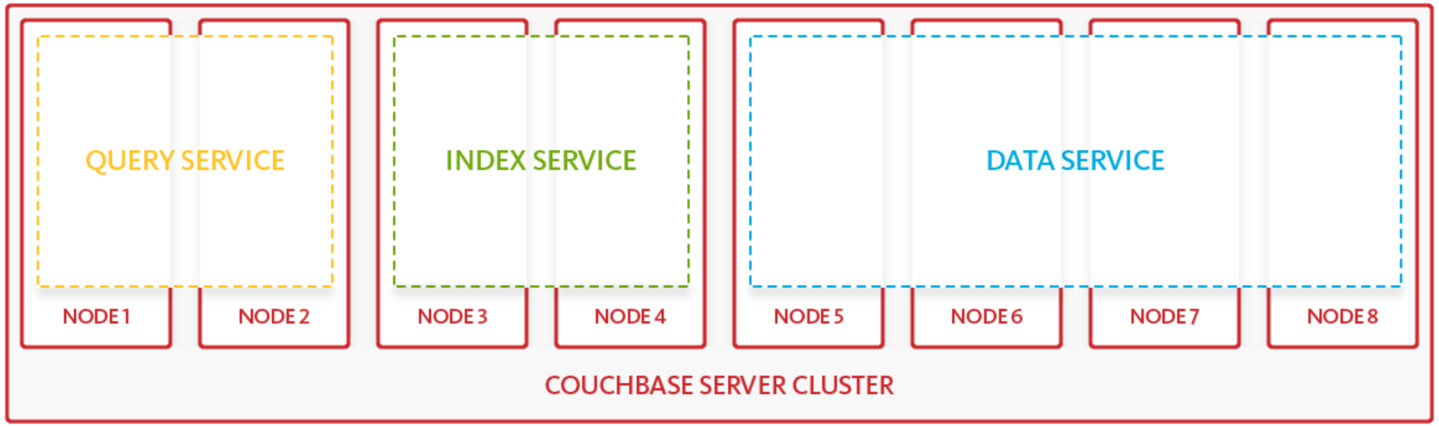
Independent Scalability for Best Computational Capacity — *per Service.*

Heavier Indexing?

Scale up or out
Index Service Nodes.

More RAM for Query Processing?

Scale up or out
Query Service Nodes.



Modern Architecture – Multi-Dimensional Scaling

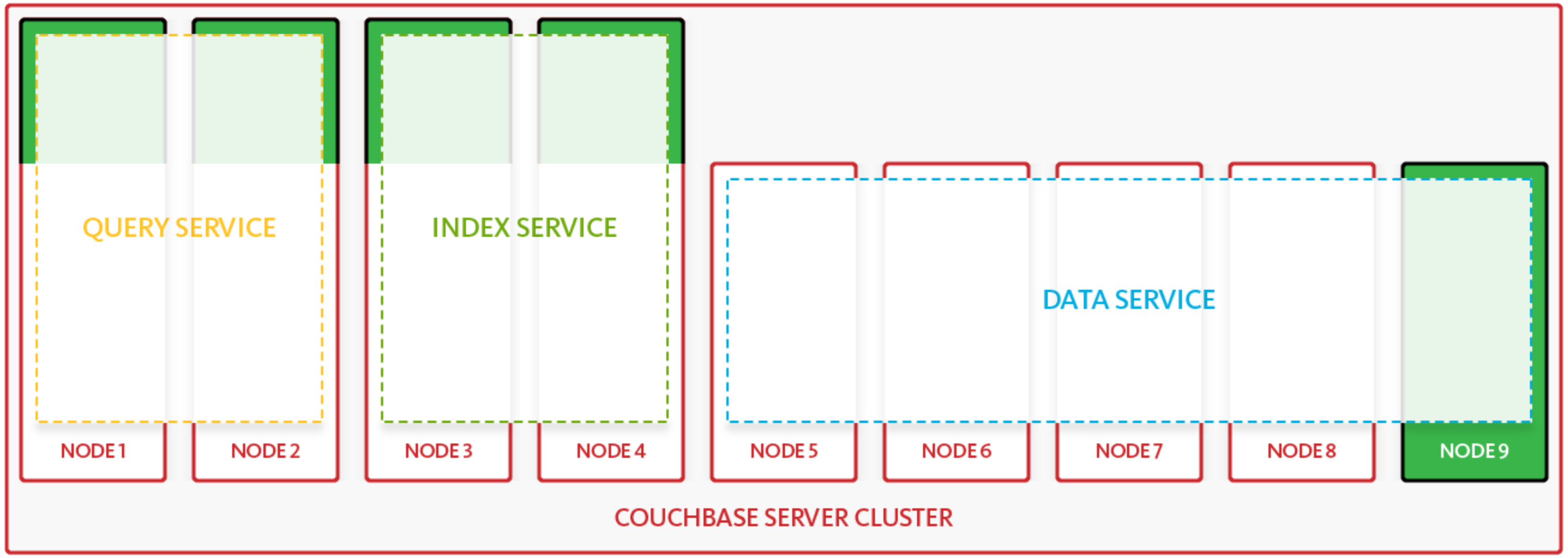
Independent Scalability for Best Computational Capacity — *per Service.*

Heavier Indexing?

Scale up or out
Index Service Nodes.

More RAM for Query Processing?

Scale up or out
Query Service Nodes.

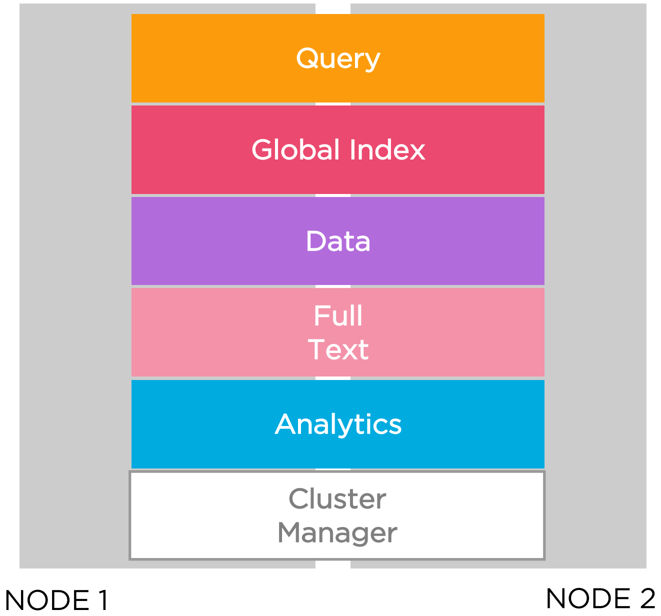


The same applies for other services

5 | Core Principles

#1 Elastic Scaling Architecture

Sample Dev Setup



Query

Global Index

Data

Full
Text

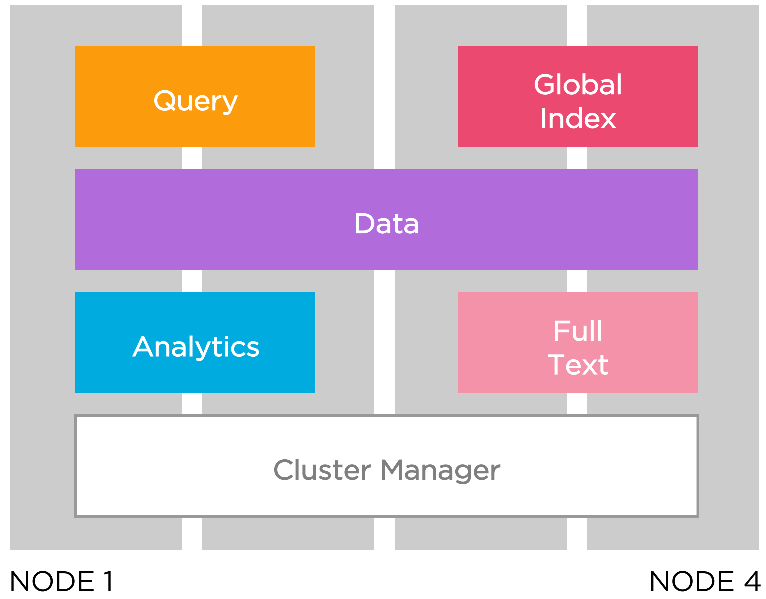
Analytics

Cluster
Manager

NODE 1

NODE 2

Sample QA Setup



Query

Global
Index

Data

Analytics

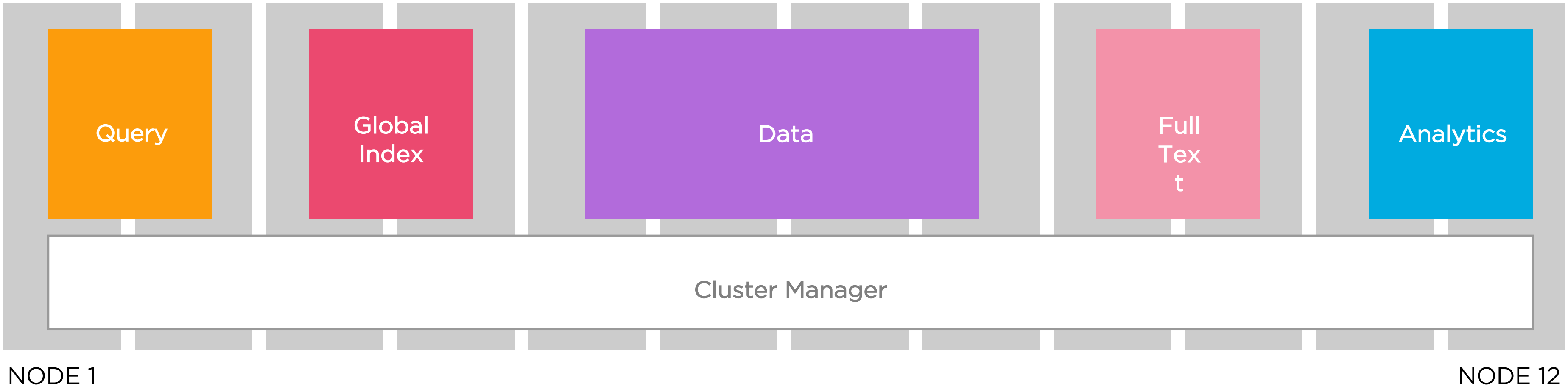
Full
Text

Cluster Manager

NODE 1

NODE 4

Sample Production Deployment



Query

Global
Index

Data

Full
Tex
t

Analytics

Cluster Manager

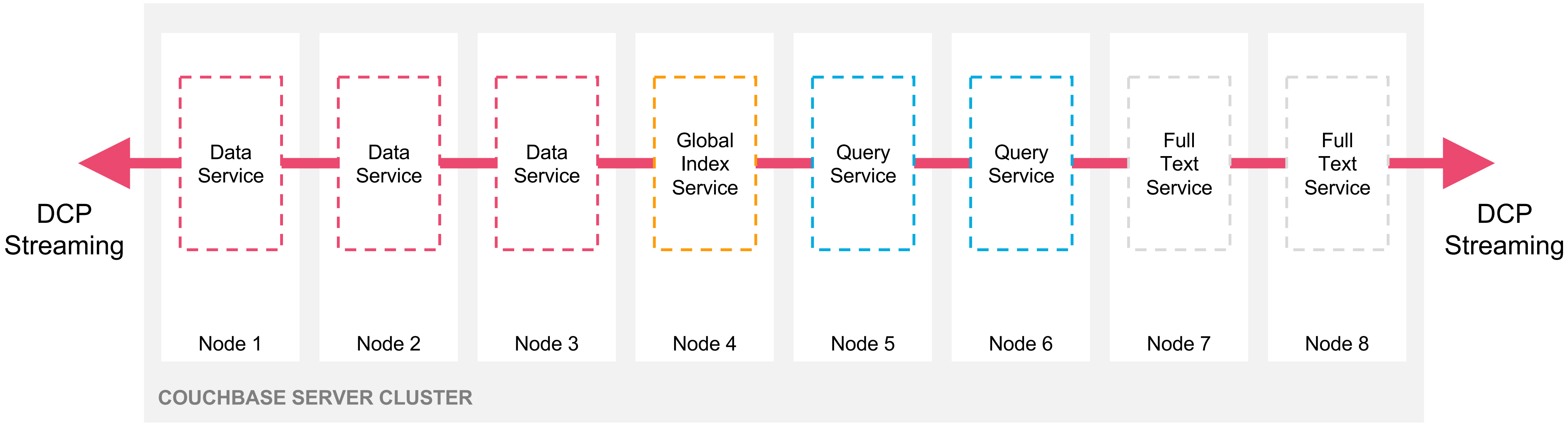
NODE 1

NODE 12

©2017 Couchbase. All rights reserved.

#2 Memory-first architecture

Data movement free from disk bottlenecks



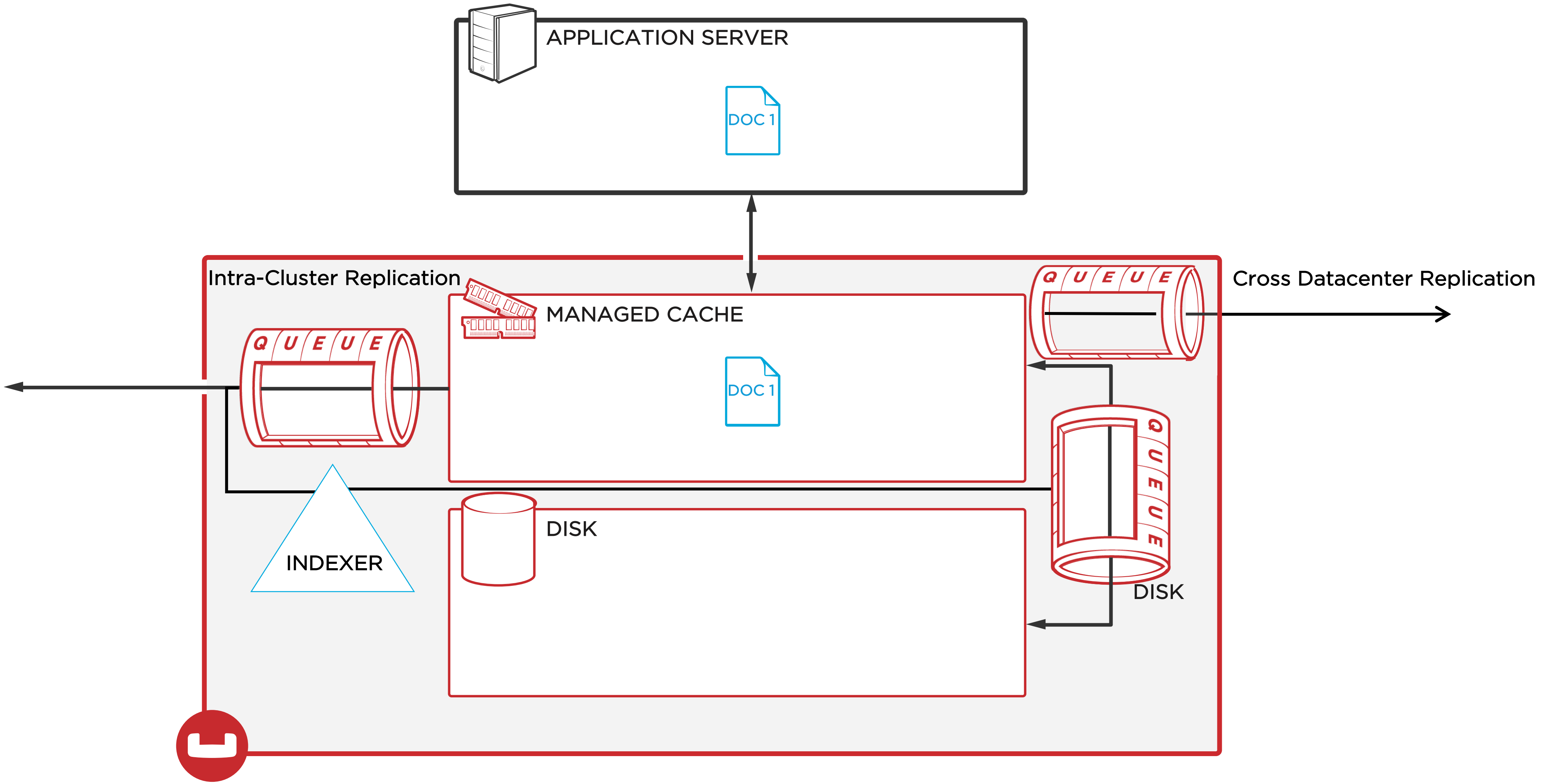
- In-memory streaming of updates to all components
- In-memory cache
- Memory-only data buckets
- Memory-only indexes

In-memory streaming of updates to all components
In-memory (cached) access to data and indexes
Memory-only indexes

#3 Asynchronous approach to everything

- Persistence
- Intra-cluster Replication
- Inter-cluster Replication
- Global secondary Indexing updates
- Full-Text Search update
- Analytics service updates

#3 Asynchronous approach to everything



#3 Asynchronous approach to everything

Configurable consistency per request / query

Data Consistency

Data access is strongly consistent within cluster
Eventually consistent across clusters

Query Consistency

Specify level of consistency for queries

Thank you

Couchbase

©2017 Couchbase. All rights reserved.