

Bucket

About Data Buckets

- Couchbase Server stores all of your application data in either RAM or on disk.
- The data containers used in Couchbase Server are called buckets
- There are Three bucket types in Couchbase, which reflect the two types of data storage that we use in Couchbase Server.
- Buckets also serve as namespaces for documents and are used to look up a document by key.

Partitioning Data with Buckets

- can partition your data into separate buckets with Couchbase Server.
- Couchbase will keep separate storage for different buckets, which enables you to perform operations such as statistics.
- Separating buckets is also a structure you may choose if you have a particular bucket that is reserved for data removal.

- Couchbase buckets
- Ephemeral buckets
- Memcached buckets

Couchbase Buckets

- provide data persistence and data replication.
- Data stored in Couchbase Buckets is highly-available
- can survive node failures
- restore data plus allow cluster reconfiguration
- Fully supports replication and server rebalancing
- Supports items up to 20MB in size.

Couchbase*/Ephemeral	Memcached
<ul style="list-style-type: none"> ✓ highly-available ✓ dynamically reconfigurable distributed data storage ✓ Persistence* ✓ replication services 	<ul style="list-style-type: none"> ✓ directly-addressed ✓ distributed (scale-out) ✓ in-memory ✓ key-value cache

Capability	Memcached buckets	Couchbase buckets	Ephemeral buckets
Item size limit	1 MB	20 MB	20 MB
Persistence	No	Yes	No
Replication (DCP)	No	Yes	Yes
Cross Datacenter Replication (XDCR)	No	Yes	Yes

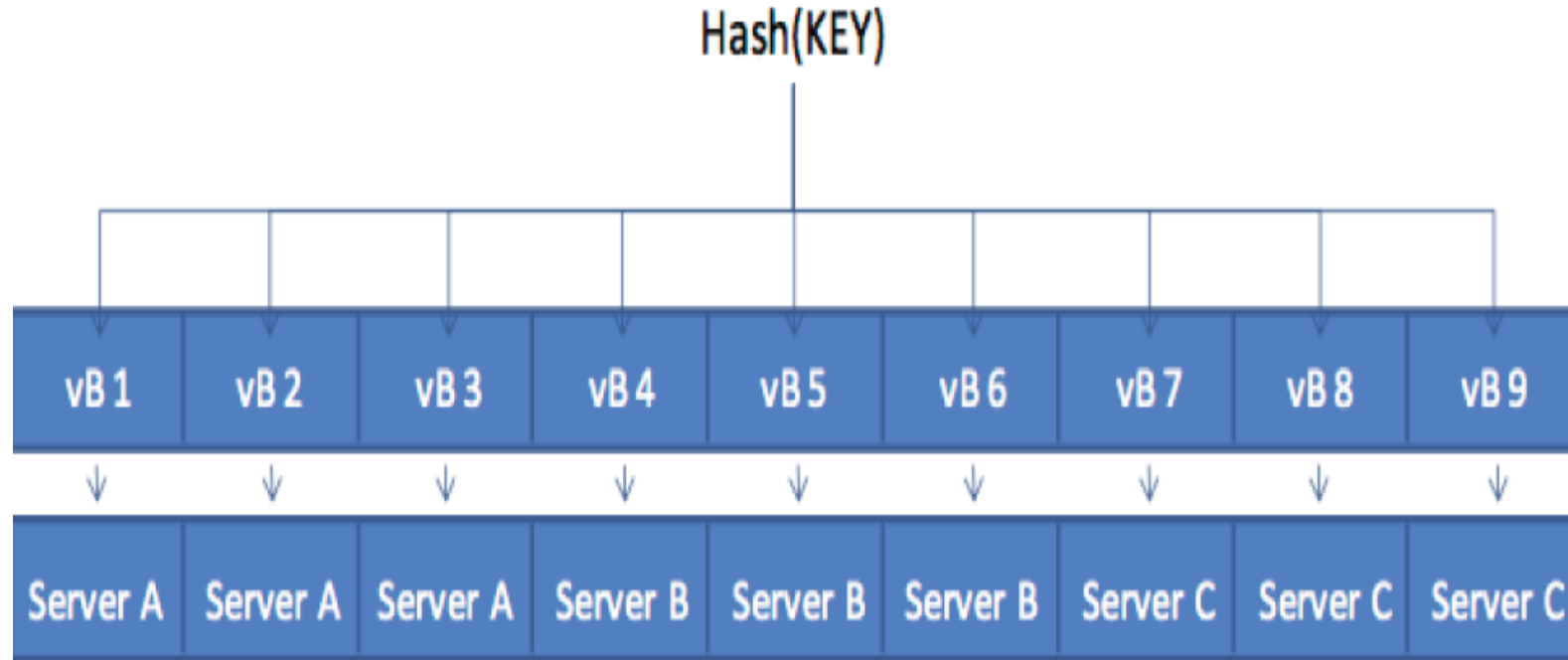
Capability	Description
Caching	Couchbase buckets operate through RAM. Data is kept in RAM and persisted down to disk.
Persistence	Data objects can be persisted asynchronously to hard-disk resources from memory to provide protection from server restarts or minor failures.
Replication	configurable number of replica servers
Rebalancing	load distribution across resources and dynamic addition or removal of buckets

Capability	Couchbase Buckets	Ephemeral Buckets
Persistence	Couchbase buckets are persisted asynchronously, from memory to disk.	Ephemeral buckets are not persisted to disk: they are retained in RAM only.
Replication (DCP and XDCR)	Couchbase buckets can be replicated across a configurable number of servers.	Ephemeral buckets can be replicated across but without being persisted to disk.
Rebalance	By means of rebalancing, the load constituted by Couchbase buckets is distributed evenly across nodes within the cluster.	Rebalancing redistributes Ephemeral buckets, exactly as it does Couchbase buckets; but without the data being persisted to disk.
Auto-failover	By default, Auto-failover starts when a node has been inaccessible for 120 seconds.	Auto-failover starts as soon as a node is inaccessible

- the owner of a subset of the key space of a Couchbase cluster
- used to allow information to be distributed effectively across the cluster
- used both for distributing data, and for supporting replicas (copies of bucket data) on more than one node.

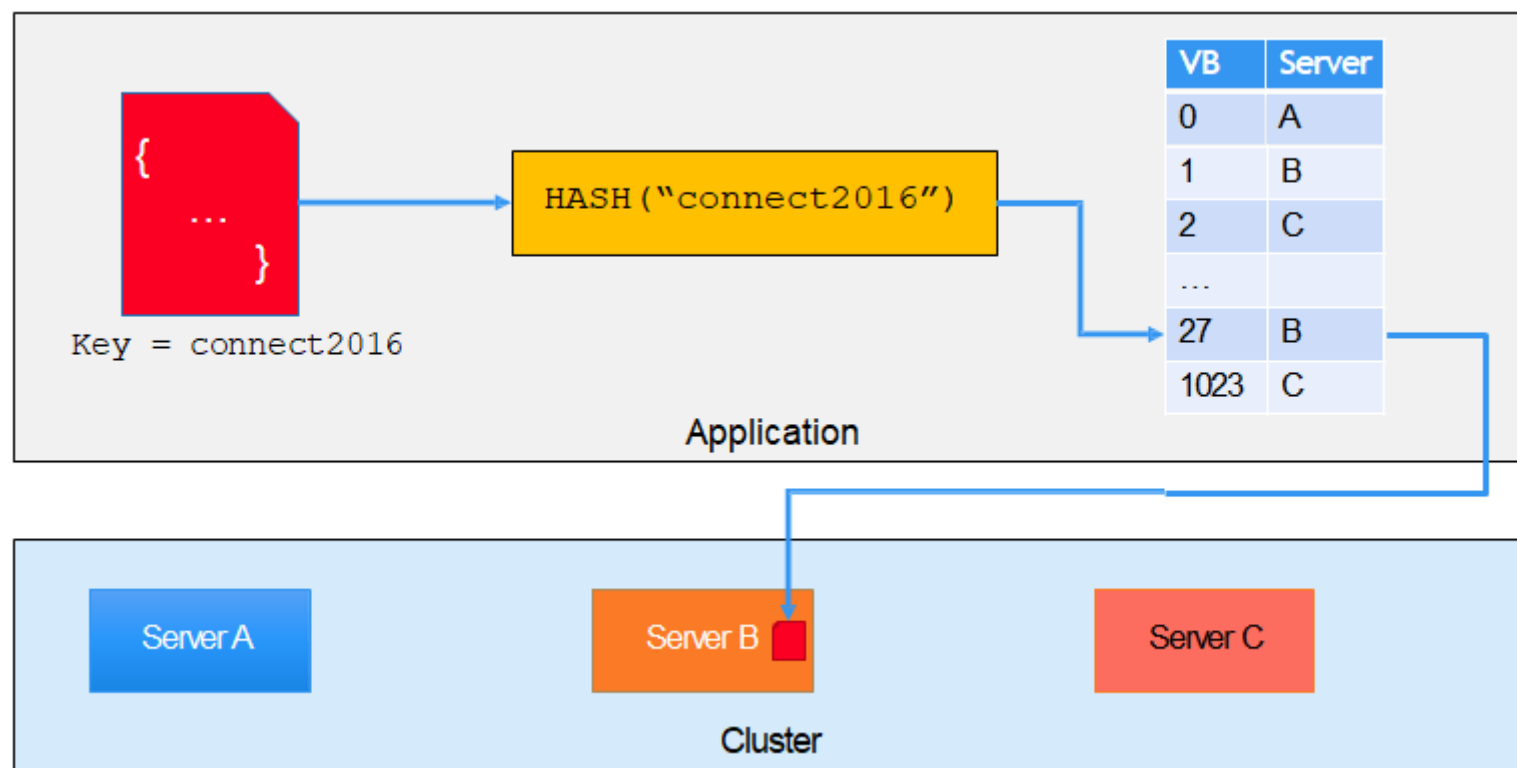
- A vBucket is defined as the *owner of a subset of the key space of a Couchbase cluster*. These vBuckets are used to allow information to be distributed effectively across the cluster.
- The vBucket system is used both for distributing data, and for supporting replicas (copies of bucket data) on more than one node.
- Clients access the information stored in a bucket by communicating directly with the node response for the corresponding vBucket. This direct access enables clients to communicate with the node storing the data, rather than using a proxy or redistribution architecture.
- The result is abstracting the physical topology from the logical partitioning of data. This architecture is what gives Couchbase Server elasticity

vBucket Mapping

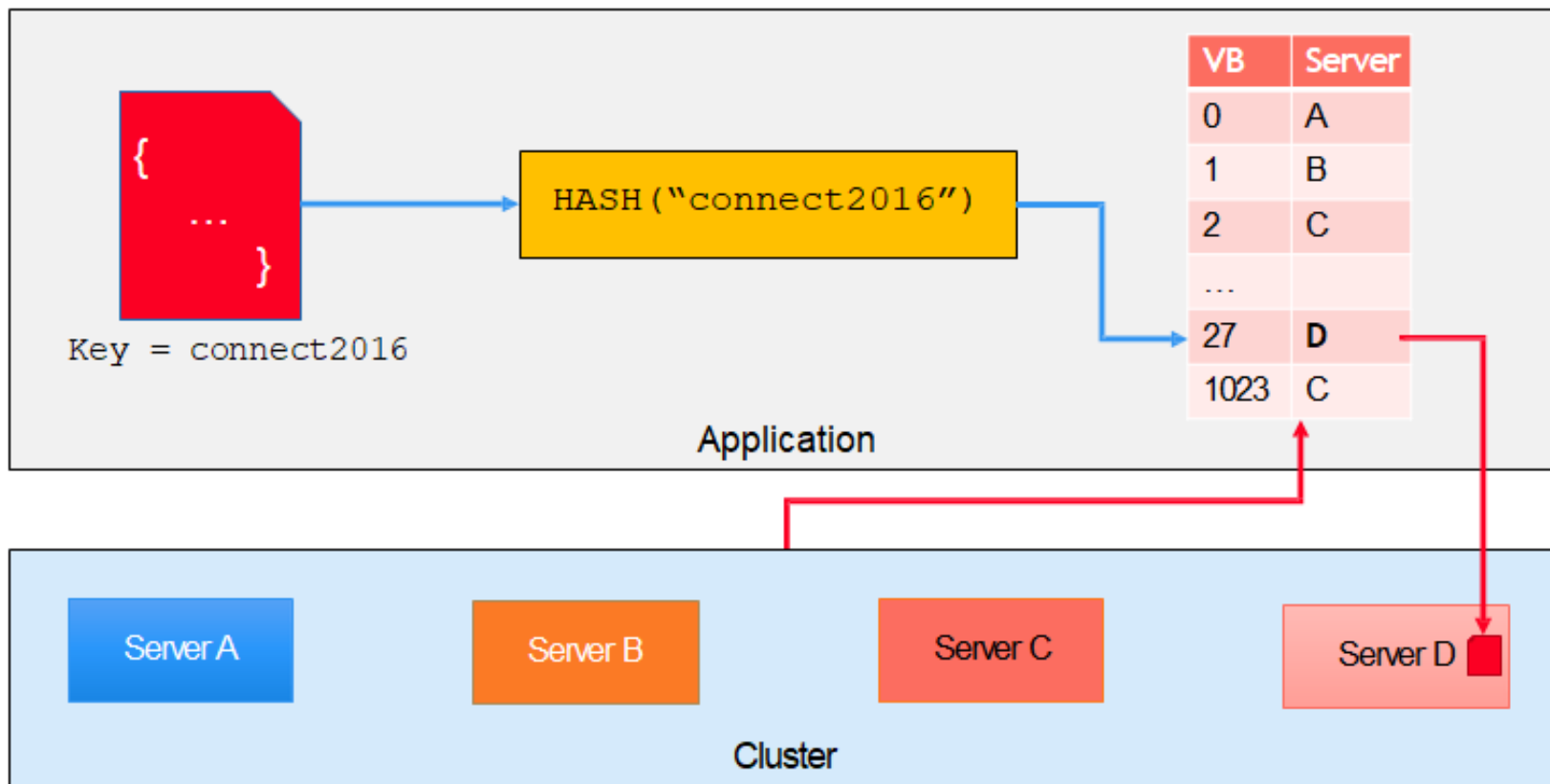


The client first hashes the key to calculate the vBucket which owns KEY.

Automatic Sharding



Automatic Sharding



[Dashboard](#)[Servers](#)[Buckets](#)[Indexes](#)[Search](#)[Query](#)[XDCR](#)[Security](#)[Settings](#)[Logs](#)

name	items	resident	ops/sec	RAM used/quota	disk used	
testBucket	1	100%	0	3.52MB / 650MB	283KB	Documents Statistics
travel-sample	31,591	100%	0	44.6MB / 100MB	21.7MB	Documents Statistics

Creating a new Bucket

Add Data Bucket

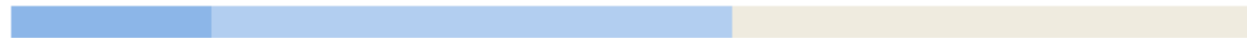


Name

Bucket name cannot be empty

Memory Quota in megabytes per server node

MB



☐ other buckets (100 MB) ☒ this bucket (250 MB) ☐ remaining (250 MB)

Bucket Type

☒ Couchbase ☐ Memcached ☐ Ephemeral

► Advanced bucket settings

Cancel





Add Bucket

Creating a new Bucket...

Add Data Bucket ✕

Name

Memory Quota in megabytes per server node
 MB


 other buckets (100 MB)  this bucket (250 MB)  remaining (250 MB)

Bucket Type
☒ Couchbase ☐ Memcached ☐ Ephemeral

▼ Advanced bucket settings

Replicas
☒ Enable Number of replica (backup) copies
☐ Replicate view indexes

Conflict Resolution ⓘ
☒ Sequence number ☐ Timestamp

Ejection Method ⓘ
☒ Value-only ☐ Full

Bucket Priority ⓘ
☒ Default ☐ High

Auto-Compaction ⓘ
☐ Override the default auto-compaction settings?

Flush ⓘ
☐ Enable

Cancel Add Bucket

Add Data Bucket ✕

Name

Memory Quota in megabytes per server node
 MB

☒ other buckets (100 MB) ☒ this bucket (250 MB) ☐ remaining (250 MB)

Bucket Type
☐ Couchbase ☐ Memcached ☒ Ephemeral

Advanced bucket settings

Replicas
☒ Enable Number of replica (backup) copies

Conflict Resolution ⓘ
☒ Sequence number ☐ Timestamp

Ejection Method ⓘ
☒ No ejection ☐ NRU ejection

Metadata Purge Interval ⓘ
 Range 0.04 (1 H) - 60days

Bucket Priority ⓘ
☒ Default ☐ High

Flush ⓘ
☐ Enable

Cancel Add Bucket

[Dashboard](#)[Servers](#)[Buckets](#)[Indexes](#)[Search](#)[Query](#)[XDCR](#)[Security](#)[Settings](#)[Logs](#)

name	items	resident	ops/sec	RAM used/quota	disk used	
mySecondTestBucket	1	100%	0	48MB / 100MB	4.06MB	Documents Statistics
myTestBucket	2	100%	0	48MB / 100MB	4.07MB	Documents Statistics

Cluster (7.0): **cb-docs**

Bucket: **travel-sample**

Scope: **inventory**

Collection: **airline**

Document: **airline_10**

```
{
  "id": 10,
  "type": "airline",
  "name": "40-Mile Air",
  "iata": "Q5",
  "icao": "MLA",
  "callsign":
  "MILE-AIR",
  "country": "United
States"
}
```

A maximum of 30 buckets can be created in a cluster

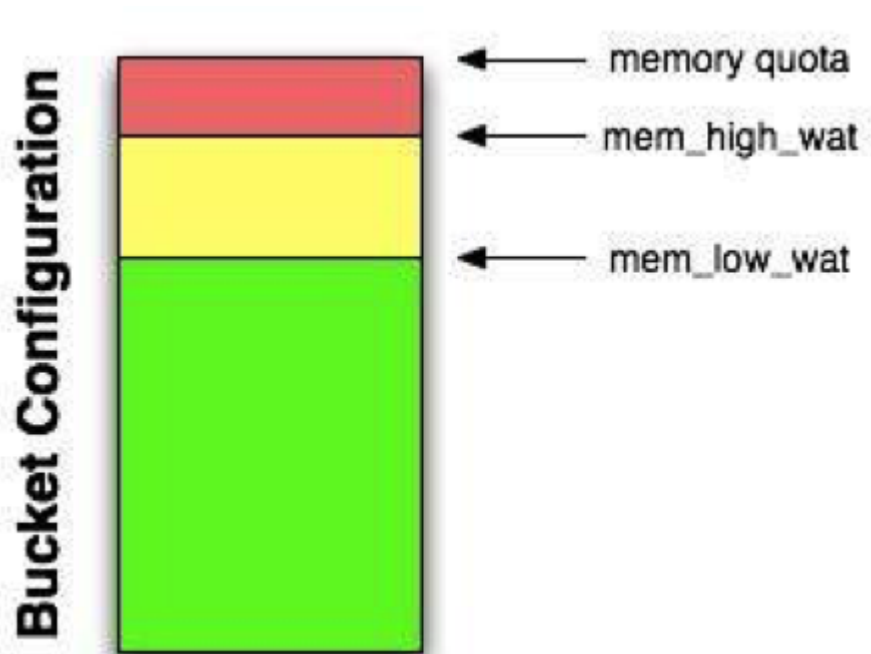
1000 collections can be created per cluster

- The benefits of scopes and collections include:
 - The logical grouping of similar documents; potentially simplifying operations such as query, XDCR, and backup and restore.
 - The increased efficiency of indexing, due to the Data Service being able to provide documents from specific collections to the Index Service.
 - Simplified querying, since query statements are able more easily to specify particular subsets of documents.
 - Easier migration from relational databases to Couchbase Server, since collections can be designed to correspond to pre-existing relational tables.
 - Secure isolation of different document-types, within a bucket; allowing applications to be specifically authorized to use only their appropriate subsets of data

- provide an extensive built-in caching layer that acts as a central part of the operation of the system.
- client interface works through the RAM-based data store

- The architecture of Couchbase Server includes a built-in caching layer. This approach allows for very fast response times, since the data is initially written to RAM by the client, and can be returned from RAM to the client when the data is requested.
- The effect of this design to provide an extensive built-in caching layer that acts as a central part of the operation of the system.
- The client interface works through the RAMbased data store, with information stored by the clients written into RAM and data retrieved by the clients returned from RAM; or loaded from disk into RAM before being returned to the client.
- For the highest performance, you should allocate the maximum amount of RAM on each of your nodes. The aggregated RAM is used across the cluster.

- process of removing data from RAM to make room for active and more frequently used information.
- automatic and operates in conjunction with the disk persistence system



Default setting for RAM water marks

Version	High water mark	Low water mark
2.0	75%	60%
2.0.1 and higher	85%	75%

- Ejection is a mechanism used with Couchbase buckets, and is the process of removing data from RAM to make room for active and more frequently used information—a key part of the caching mechanism. Ejection is automatic and operates in conjunction with the disk persistence system to ensure that data in RAM has been persisted to disk and can be safely ejected from the system.
- The system ensures that the data stored in RAM will already have been written to disk, so that it can be loaded back into RAM if the data is requested by a client. Ejection is a key part of keeping frequently used information in RAM and ensuring that there is space within the Couchbase RAM allocation to load that data back into RAM when the information is requested by a client.

- Used for data with a naturally limited life.
- to be automatically deleted from the entire database.
- expiration time :
 - as a relative time (for example, in 60 seconds),
 - or absolute time (31st December 2012, 12:00 p.m.)

- Each document stored in the database has an optional expiration value. The default is for there to be no expiration (i.e., the information will be stored indefinitely). The expiration can be used for data with a naturally limited life that you want to be automatically deleted from the entire database.
- The expiration value is user-specified on a document basis at the point when the data is stored. The expiration can also be updated when the data is updated, or explicitly changed through the Couchbase protocol. The expiration time can either be specified as a relative time (for example, in 60 seconds), or absolute time (31st December 2012, 12:00 p.m.).
- Typical uses for an expiration value include web session data, where you want the actively stored information to be removed from the system if the user activity has stopped and not been explicitly deleted. The data will time out and be removed from the system, freeing up RAM and disk for more active data.

- removing information entirely from memory for Memcached buckets.
- uses a least recently used (LRU) algorithm.

- Eviction is the process of removing information entirely from memory for Memcached buckets.
- The Memcached system uses a least recently used (LRU) algorithm to remove data from the system entirely when it is no longer used.

Lab : Bucket , Scope and Collections– 60 Minutes (D)