NAME

cbc – Couchbase command line utility

SYNOPSIS

cbc help

cbc cat [**common options**] [**-r**] *key...*

cbc cp [**common options**] [**-p**] [**-r** *num*] [**-m** *num*] [**-j**] *filename...*

cbc create [**common options**] [**-f** *flag*] [**-e** *exptime*] [**-a**] *key*

cbc observe [**common options**] *key*

cbc flush [**common options**]

cbc hash [**common options**] *key...*

cbc lock [**common options**] [**-e** *exptime*] *key...*

cbc unlock [**common options**] *key cas...*

cbc rm [**common options**] *key...*

cbc stats [**common options**] [**stat group**]...

cbc verify [**common options**] *key...*

cbc version [**common options**]

cbc verbosity [**common options**] *level* [**server**]...

cbc view [**common options**] [**-c**] [**-d** *data*] [**-X** *method*] *query*

cbc admin [**common options**] [**-c**] [**-d** *data*] [**-X** *method*] *query*

cbc bucket-create [**common options**] [**-B** *type*] [**-q** *quota*] [**-a** *auth*] [**-s** *sasl-password*] [**-r** *replicas*] [**-p** *port*] *bucketname*

cbc bucket-delete [**common options**] *bucket...*

cbc bucket-flush [**common options**] *bucket...*

DESCRIPTION

cbc is a command line utility that allows you to communicate with your Couchbase Server cluster from the command line prompt.

COMMON OPTIONS

The following options are supported:

-h --help

Display usage information and exit.

-h --host

Specify the list of hosts to connect to (default: "127.0.0.1:8091").

-b --bucket

Specify the bucket to use (default: "default").

-u --user

Specify the username used for authentication to the cluster.

-P --password

Specify the password used for authentication to the cluster.

-T --enable-timings

Enable the recording of the timing of commands.

-t --timeout

Specify timeout value.

SUBCOMMANDS

The following subcommands are supported:

cbc help

Display usage information and exit.

cbc cat [**common options**] [**-r**] *key...*

Print the contents of the value for the key to standard output.

-r --replicas

Use one of the replicas instead of the master server.

cbc cp [**common options**] [**-p**] [**-r** *num*] [**-m** *num*] [**-j**] *filename...*

Store the content of a file under the specified key in the cluster.

-p --persisted

Ensure that key has been persisted to the primary node.

-r *num* **--replicated**=*num*

Ensure that the key has been replicated and persisted to given number of replicas.

-m *num* **--max-tries**=*num*

The number of attempts for observing keys (default: 5).

-j --json

Treat the value as a JSON document (take key from '_id' attribute). This option is only valid if the libyajl2 is present.

cbc create [**common options**] [**-f** *flag*] [**-e** *exptime*] [**-a**] *key*
Create a key in the cluster by reading the the value from standard input.

    -f *value* **--flag**=*value*

        The flags to associate with the key.

    -e *value* **--exptime**=*value*

        The expiration time for the key.

    -a

        Fail if an object exist in the database for that key.

cbc observe [**common options**] *key*
Observe a key in the cache.

cbc flush [**common options**]
Remove all keys from the cluster. The flush subcommand is only supported on memcached buckets. To flush a Couchbase bucket you need use **bucket-flush**.

cbc hash [**common options**] [**-f** *filename*] *key...*
hash key(s) and print out useful info.

    -f *filename* **--config-file**=*filename*

        filename shall be a plain text file containing an alternative cluster configuration (in JSON) to use.

cbc lock [**common options**] [**-e** *exptime*] *key...*
Lock and retrieve the value for a key. The lock is held for the object until it expires (timing out) or from a manual unlock command. Consult your Couchbase documentation for more information about locking of objects.

    -e *value* **--exptime**=*value*

        The expiry time for the lock.

cbc unlock [**common options**] *key cas...*
Unlock the key previously locked with lock. You have to specify the same cas value as returned by the lock command in order to successfully unlock the keys.

cbc rm [**common options**] *key...*
Remove a number of keys from the cluster.

cbc stats [**common options**] [**stat group**]...
    Retrieve various statistics from the cluster.

cbc verify [**common options**] *filename*...
    Verify the content for the key represented by the filename in the cache is the same as the file content.

cbc version [**common options**]
    Print the version numbers for cbc and libcouchbase.

<div align="center">

*R [***server***]...*
</div>

cbc verbosity [**common options**] *level*
    Set verbosity level. The level may be one of the following:

    detail

        This will cause the nodes to generate *an insane* amount of data. It shoud not be used unless you
        know what you're doing.

    debug

        This will cause the nodes to generate *a lot* of data. It should not be used unless you know what
        you're doing.

    info

        This will cause the nodes to generate *lot* of data (dumping each command being executed). You
        should avoid using this unless you're searching for a bug. It will affect your performance.

    warning

        Only warnings will be reported. This is what you normally want!

cbc view [**common options**] [**-c**] [**-d** *data*] [**-X** *method*] *query*
    Execute Couchbase view (aka map/reduce) request.

cbc admin [**common options**] [**-c**] [**-d** *data*] [**-X** *method*] *query*
    execute request to management REST API.

cbc bucket-create [**common options**] [**-B** *type*] [**-q** *quota*] [**-a** *auth*] [**-s** *sasl-password*] [**-r** *replicas*] [**-p**
port] *bucketname*
    Create a bucket in the Cluster.

    -B *type* **--bucket-type**=*type*

        Specify the type of bucket to create. Type may be one of "couchbase", "memcached"

-q *value* **--ram-quota**=*value*

>   RAM quota in megabytes.

-a *type* **--auth-type**=*type*

>   Type of bucket authentication, type may be one of "none" or "sasl".

-s *passwd* **--sasl-password**=*passwd*

>   Password used for sasl authentication.

-r *num* **--replica-number**=*num*

>   The number of replicas to create for each key. The value should be in the range [0-3].

-p *port* **--proxy-port**=*port*

>   The port number the proxy should provide access to this bucket.

cbc bucket-delete [**common options**] *bucket...*
>   Delete the named buckets from the cluster.

cbc bucket-flush [**common options**] *bucket...*
>   Flush (remove all data) from the named buckets. Please note that you need to have flush enabled on
>   the specified bucket to use this command successfully.

EXAMPLES

>   **Example 1** Copy a file into the cluster
>
>   The following command copies the file mynote.txt located in the current directory into the cluster:
>
>
>   example$ **cbc cp mynote.txt**
>   Stored "mynote.txt" CAS:d8062155b1100000
>
>
>   **Example 2** Observe a key in the cluster
>
>   The following command retrieves information about the key named mynote.txt:
>
>   example$ **cbc observe mynote.txt**
>   PERSISTED "mynote.txt" CAS:313e468316000000 IsMaster:true TimeToPersist:0 TimeToReplicate:0

Example 3 cbc hash

The following command shows you how to use **cbc hash**:

example$ **cbc hash key1 key2 key3**
"key1"   vBucket:92 Server:"127.0.0.1:12000" CouchAPI:"http://127.0.0.1:9500/default" Replicas:"127.0.0.1:12000"
"key2"   vBucket:341 Server:"127.0.0.1:12000" CouchAPI:"http://127.0.0.1:9500/default" Replicas:"127.0.0.1:12000"
"key3"   vBucket:594 Server:"127.0.0.1:12000" CouchAPI:"http://127.0.0.1:9500/default" Replicas:"127.0.0.1:12000"

Example 4 Create a bucket

The following command shows you how to create a bucket in the cache. This is a privileged operation so
you need to authenticate to the cluster:

example$ **cbc bucket-create -u Administrator -P secret --bucket-type=memcached --ram-quota=64 --auth-type=sas**
Server: Couchbase Server 2.0.0r_521_g67b4898
Pragma: no-cache
Location: /pools/default/buckets/mybucket
Date: Tue, 06 Nov 2012 11:04:40 GMT
Content-Length: 0
Cache-Control: no-cache
"/pools/default/buckets": OK Size:0

Example 5 Flush a bucket

The following command shows you how to flush (remove all items) in the bucket named "mybucket":

example$ **cbc bucket-flush mybucket**
Server: Couchbase Server 2.0.0r_521_g67b4898
Pragma: no-cache
Date: Tue, 06 Nov 2012 11:12:33 GMT
Content-Length: 0
Cache-Control: no-cache
"/pools/default/buckets/mybucket/controller/doFlush": OK Size:0

Example 6 Delete a bucket

The following command shows you delete the bucket named "mybucket". This is a privileged operation so
you need to authenticate to the cluster:

example$ **cbc bucket-delete -u Administrator -P secret --timeout=10000000 mybucket**
Server: Couchbase Server 2.0.0r_521_g67b4898
Pragma: no-cache
Date: Tue, 06 Nov 2012 11:25:57 GMT
Content-Length: 0
Cache-Control: no-cache
"/pools/default/buckets/mybucket": OK Size:0

FILES

      ˜/.cbcrc

           Default values used by cbc. See **cbcrc**(4) for more information

ENVIRONMENT VARIABLES

      The following environment variables may be used to specify configuration values. If specified they override the value specified in **˜/.cbcrc** (but options specified on the command line will override environment variables).

      COUCHBASE_CLUSTER_URI

           This is a list separated by semicolon of hostnames (with an optional port) to your cluster.

      COUCHBASE_CLUSTER_USER

           This is the username used during authentication to your cluster.

      COUCHBASE_CLUSTER_PASSWORD

           This is the password used during authentication to your cluster.

      COUCHBASE_CLUSTER_BUCKET

           This is the name of the bucket you would like to use.

ATTRIBUTES

      See **attributes**(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Volatile

SEE ALSO

      cbcrc(4), **attributes**(5)