


3 shards

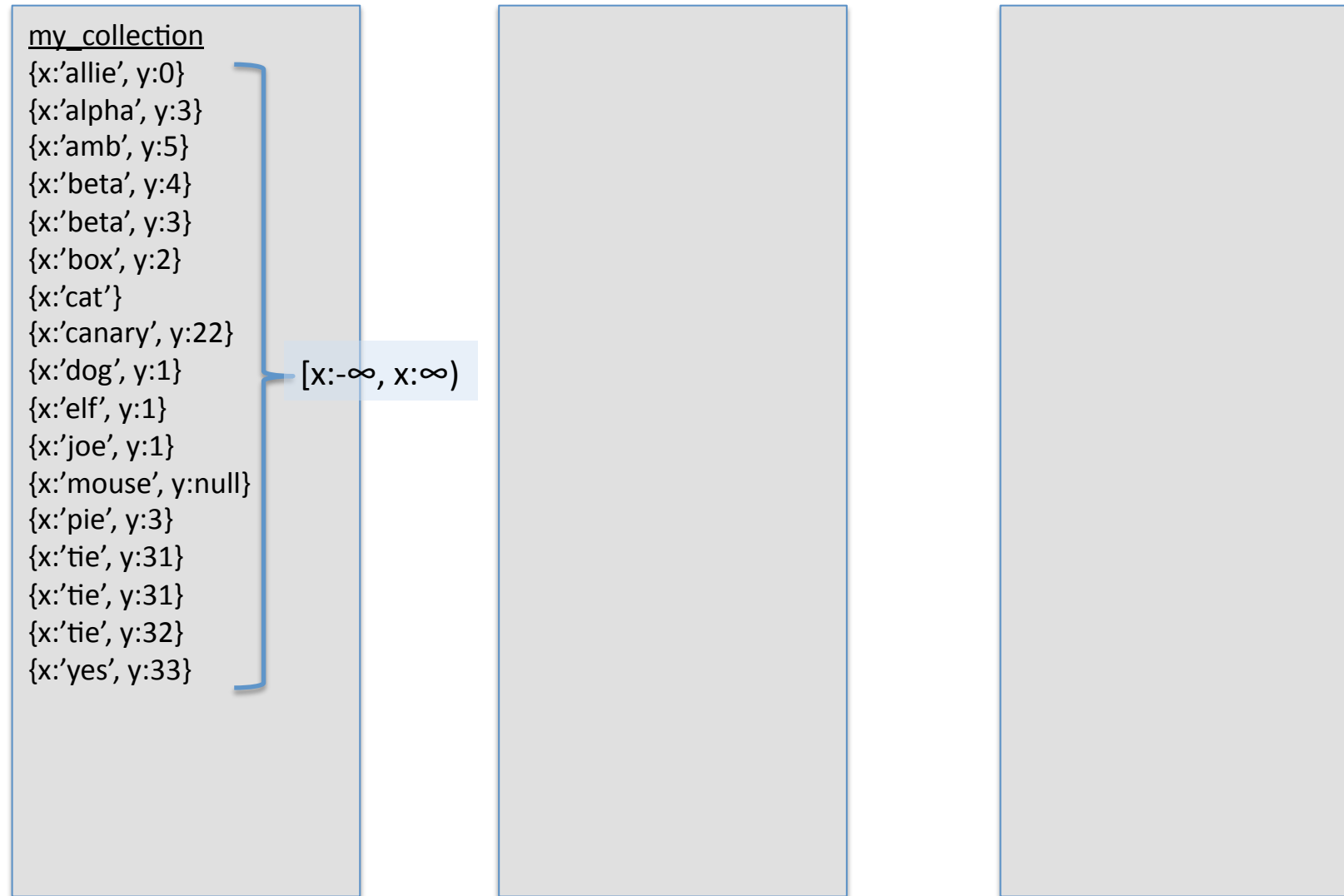
my_collection
{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}
{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'pie', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

we shard "my_collection" by the shardkey pattern { x : 1 }

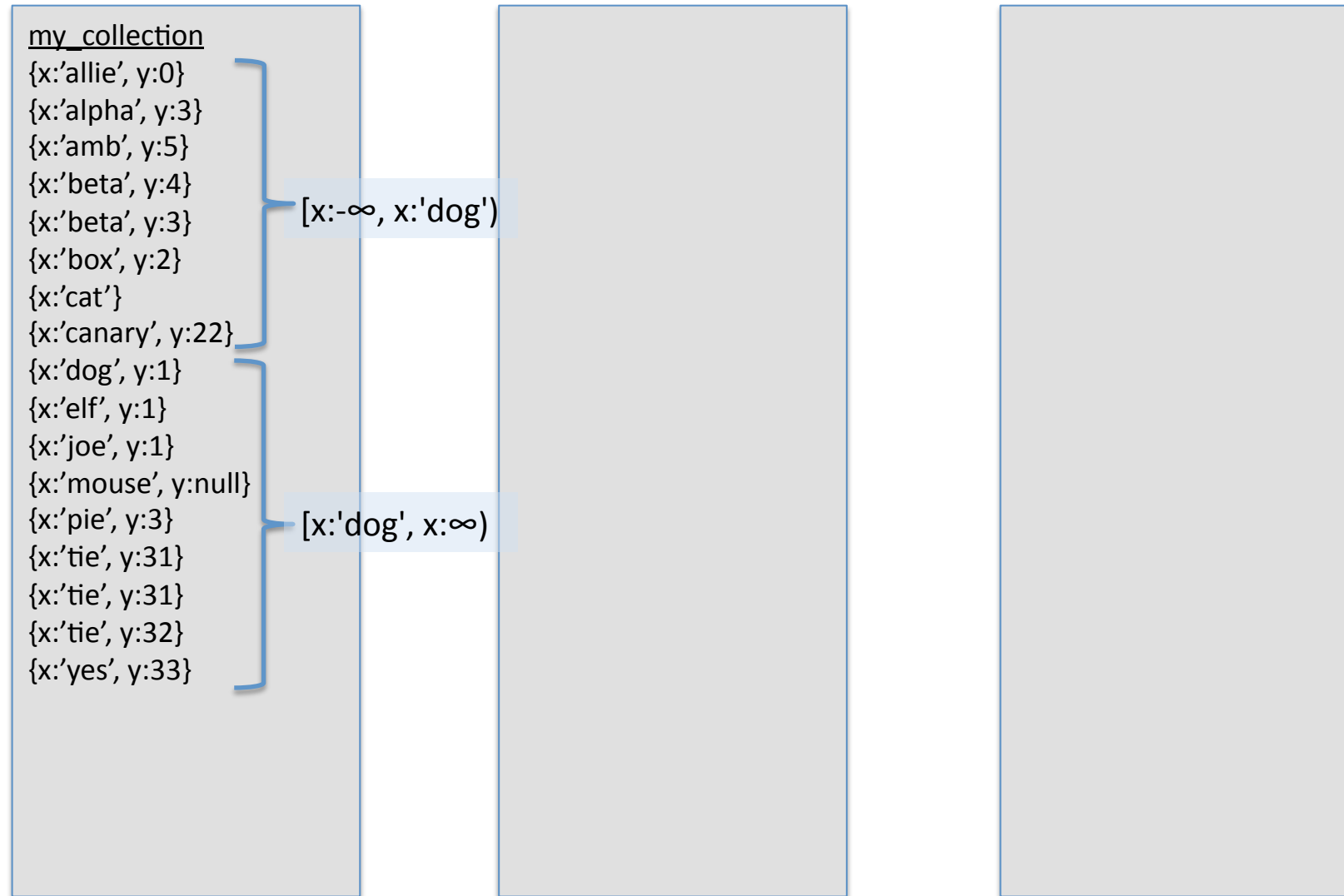


my_collection
{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}
{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'pie', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

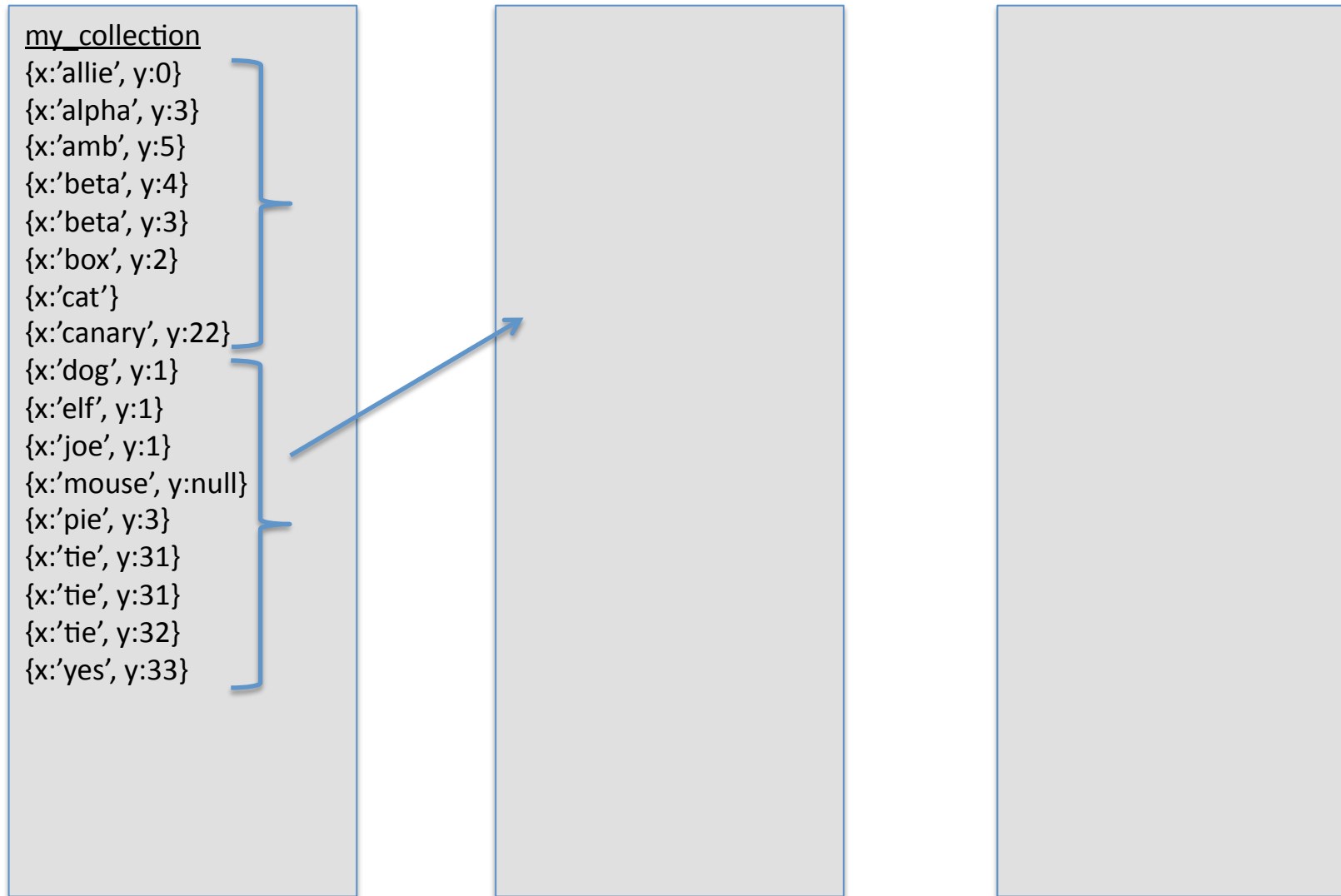
at first there is a single *chunk* for all data for this one collection. all data in the range is on the first shard.



when a chunk exceeds approximately 200MB, it *splits*. we now have 2 chunks.



the *balancer* will *migrate* a chunk from one shard to the other in the background.



documents with shard key values which are nearby in ordering tend to be in the same chunk, and thus on the same server.

my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}

my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'pie', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

here we have added new data where x values begin with 'p', showing a case where the shard key distribution is quite non-uniform.

my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}


my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'paa', y:3}
{x:'pab', y:3}
{x:'pac', y:3}
{x:'pad', y:3}
{x:'pae', y:3}
{x:'pf', y:3}
{x:'pg', y:3}
{x:'pie', y:3}
{x:'pqe', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

a split...


my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}



my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'paa', y:3}
{x:'pab', y:3}
{x:'pac', y:3}
{x:'pad', y:3}
{x:'pae', y:3}
{x:'pf', y:3}
{x:'pg', y:3}
{x:'pie', y:3}
{x:'pqe', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}



a migrate...

my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}

my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'paa', y:3}
{x:'pab', y:3}
{x:'pac', y:3}
{x:'pad', y:3}
{x:'pae', y:3}
{x:'pf', y:3}
{x:'pg', y:3}
{x:'pie', y:3}
{x:'pqe', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}



after the migrate

my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}

my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'paa', y:3}
{x:'pab', y:3}
{x:'pac', y:3}
{x:'pad', y:3}

my_collection

{x:'pae', y:3}
{x:'pf', y:3}
{x:'pg', y:3}
{x:'pie', y:3}
{x:'pqe', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

metadata is maintained for each chunk (shown in inset below). this metadata is stored in the cluster's *config servers*.

my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}

my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'paa', y:3}
{x:'pab', y:3}
{x:'pac', y:3}
{x:'pad', y:3}

my_collection

{x:'pae', y:3}
{x:'pf', y:3}
{x:'pg', y:3}
{x:'pie', y:3}
{x:'pqe', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

chunks

[x:-∞, x:'dog') -> S1
[x:'dog', x:'pae') -> S2
[x:'pae', x:∞) -> S3

my_collection

{x:'allie', y:0}
{x:'alpha', y:3}
{x:'amb', y:5}
{x:'beta', y:4}
{x:'beta', y:3}
{x:'box', y:2}
{x:'cat'}
{x:'canary', y:22}
{x:'na'}
{x:'nb'}
{x:'nd'}
{x:'nd'}
{x:'nq'}

my_collection

{x:'dog', y:1}
{x:'elf', y:1}
{x:'joe', y:1}
{x:'mouse', y:null}
{x:'paa', y:3}
{x:'pab', y:3}
{x:'pac', y:3}
{x:'pad', y:3}

my_collection

{x:'pae', y:3}
{x:'pf', y:3}
{x:'pg', y:3}
{x:'pie', y:3}
{x:'pqe', y:3}
{x:'tie', y:31}
{x:'tie', y:31}
{x:'tie', y:32}
{x:'yes', y:33}

chunks

[x:-∞, x:'dog') -> S1
[x:'dog', x:'na') -> S2
[x:'na', x:'paa') -> S1
[x:'paa', x:'pae') -> S2
[x:'pae', x:∞) -> S3

here we see the state of the system after several further operations, including some document insertions, splits, and a migrate (of the chunk with the 'n' x values). note that the document shard keys in a given chunk are always within a given range, but that chunks may live on any one (and only one) shard : there is no ordering of the chunks, simply an attempt to balance them among the shards.

the balancer, running in the background, works to keep the number of chunks per shard even.

additional shards can be added at any time. migration of data to the new shard will be done gradually by the balancer.