Sample Couchbase Cluster Health Check Report

Customer Name: ABC Inc.

Date created: 07-05-2020

Date delivered: 07-07-2020

Created by: Couchbase Center of Excellence

Delivered by: Couchbase Center of Excellence

Cluster Health Check Report

Cluster Overview

This Couchbase cluster has total 7 nodes. Index service is running on 2 nodes. Data service is running on 5 nodes. Query service is running on 2 nodes.

This health check report is based on analysis of Couchbase Server logs that were gathered at the times shown below.

Node Host Name	Log Collection Time
node199	2020-03-10 08:39:12
node204	2020-03-10 08:40:12
node207	2020-03-10 08:39:40
node217	2020-03-10 08:39:40
node218	2020-03-10 08:39:42
node235	2020-03-10 08:39:38
node238	2020-03-10 08:39:14

OS name and/or version doesn't match on all nodes. Read node profiles section for details.
All nodes are running the same Couchbase Server release.
Number of CPU cores doesn't match on all nodes. Read node profiles section for details.
Amount of RAM doesn't match on all nodes. Read node profiles section for details.

Note: Views are in use on this cluster. They result in longer rebalance times and very inefficient use of critical resources. Migrating to better alternatives such as N1QL, Search or Analytics is strongly recommended.

High level cluster configuration

Auto-compaction	30% (default)
Auto-failover	600 seconds
External Authentication	Enabled
Server Group 1	7 node(s): node{199, 204, 207, 217, 218, 235, 238}
Index Service	2 node(s): node{199, 238}
Data Service	5 node(s): node{204, 207, 217, 228, 235}
Query Service	2 node(s): node{199, 238}

Per node connections

	Established		TIME_WAIT		Total	
Connection	**Connections**	**Hosts**	**Connections**	**Hosts**	**Connections**	**Hosts**
node199:						
Port 8091 (cluster mgmt)	32	3	16	2	49	4
Port 8093 (N1QL)	6	3	0	0	7	4
Port 9100 (Index Admin)	2	2	0	0	3	3
Port 9101 (Index Scan)	2	2	0	0	3	3
Port 9102 (Index HTTP)	3	3	0	0	4	4
Port 9105 (Index Maintenance)	5	5	0	0	6	6

Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18093 (N1QL SSL)	0	0	0	0	1	1
==Total==	189	10	23	4	239	14
node204:						
Port 8091 (cluster mgmt)	12	4	8	1	21	5
Port 8092 (views)	0	0	0	0	1	1
Port 9999 (Projector)	0	0	0	0	1	1
Port 11207 (KV SSL)	0	0	0	0	2	2
Port 11209 (ERL -> memcached)	40	6	0	0	42	8
Port 11210 (KV)	63	5	0	0	65	7
Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18092 (views SSL)	0	0	0	0	1	1
==Total==	197	10	15	3	233	13
node207:						
Port 8091 (cluster mgmt)	10	3	0	0	11	4
Port 8092 (views)	0	0	0	0	1	1
Port 9999 (Projector)	0	0	0	0	1	1
Port 11207 (KV SSL)	0	0	0	0	2	2
Port 11209 (ERL -> memcached)	37	6	0	0	39	8
Port 11210 (KV)	792	5	0	0	794	7
Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18092 (views SSL)	0	0	0	0	1	1
==Total==	918	10	11	1	949	13
node217:						
Port 8091 (cluster mgmt)	12	3	0	0	13	4
Port 8092 (views)	0	0	0	0	1	1
Port 9999 (Projector)	0	0	0	0	1	1
Port 11207 (KV SSL)	0	0	0	0	2	2
Port 11209 (ERL -> memcached)	37	6	0	0	39	8
Port 11210 (KV)	63	5	0	0	65	7
Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18092 (views SSL)	0	0	0	0	1	1
==Total==	191	10	6	1	218	13
node218:						
Port 8091 (cluster mgmt)	10	3	0	0	11	4
Port 8092 (views)	0	0	0	0	1	1
Port 9999 (Projector)	0	0	0	0	1	1
Port 11207 (KV SSL)	0	0	0	0	2	2
Port 11209 (ERL -> memcached)	37	6	0	0	39	8
Port 11210 (KV)	63	5	0	0	65	7
Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18092 (views SSL)	0	0	0	0	1	1
==Total==	189	10	6	1	216	13
node235:						
Port 8091 (cluster mgmt)	12	3	0	0	13	4
Port 8092 (views)	0	0	0	0	1	1
Port 9999 (Projector)	0	0	0	0	1	1
Port 11207 (KV SSL)	0	0	0	0	2	2
Port 11209 (ERL -> memcached)	37	6	0	0	39	8
Port 11210 (KV)	62	5	0	0	64	7
Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18092 (views SSL)	0	0	0	0	1	1

==Total==	190	10	6	1	217	13
node238:						
Port 8091 (cluster mgmt)	27	2	8	1	36	3
Port 8093 (N1QL)	5	2	0	0	6	3
Port 9100 (Index Admin)	2	2	0	0	3	3
Port 9101 (Index Scan)	2	2	0	0	3	3
Port 9102 (Index HTTP)	3	3	0	0	4	4
Port 9105 (Index Maintenance)	5	5	0	0	6	6
Port 18091 (cluster mgmt SSL)	0	0	0	0	1	1
Port 18093 (N1QL SSL)	0	0	0	0	1	1
==Total==	177	10	14	2	218	13

Open file information

Process	File Descriptors	MMaps
node199:		
beam.smp	140	48
indexer	47	16
cbq-engine	98	8
==Total==	353	124
node204:		
beam.smp	191	52
memcached	145	43
moxi	0	0
goxdcr	12	3
projector	21	6
==Total==	378	108
node207:		
beam.smp	182	55
memcached	871	45
moxi	0	0
goxdcr	12	3
projector	21	8
==Total==	1095	115
node217:		
beam.smp	186	52
memcached	142	43
moxi	0	0
goxdcr	12	3
projector	21	6
==Total==	370	108
node218:		
beam.smp	180	52
memcached	142	43
moxi	0	0
goxdcr	12	3
projector	21	6
==Total==	364	108
node235:		
beam.smp	186	52
memcached	141	43
moxi	0	0

goxdcr	12	3
projector	21	6
==Total==	369	108
node238:		
beam.smp	134	44
indexer	55	14
cbq-engine	88	6
==Total==	345	114

Clients connected

Client	Port#	Connections	Last Connection	Unique IPs
node204:				
couchbase-net-sdk/2.6.1.0	11210	394666	2020-03-10T08:21:02	4
couchbase-net-sdk/2.7.0.0	11210	56	2020-03-03T22:05:04	2
GoMemcached	11210	1	2020-02-19T12:23:04	1
proxy	11209	4	2020-03-06T18:16:39	2
regular	11209	5	2020-03-08T05:35:41	1
node207:				
couchbase-net-sdk/2.6.1.0	11210	258232	2020-03-10T08:21:02	4
couchbase-net-sdk/2.7.0.0	11210	40	2020-03-06T17:34:20	2
GoMemcached	11210	19	2020-03-08T07:45:34	2
proxy	11209	16	2020-03-06T18:16:40	5
node217:				
couchbase-net-sdk/2.6.1.0	11210	402694	2020-03-10T08:21:02	4
couchbase-net-sdk/2.7.0.0	11210	56	2020-03-03T22:04:56	2
GoMemcached	11210	1	2020-02-21T22:15:14	1
proxy	11209	4	2020-03-06T18:16:40	2
regular	11209	3	2020-03-04T22:43:42	1
node218:				
couchbase-net-sdk/2.6.1.0	11210	394726	2020-03-10T08:21:02	4
couchbase-net-sdk/2.7.0.0	11210	56	2020-03-03T22:04:59	2
GoMemcached	11210	1	2020-02-26T01:30:33	1
proxy	11209	4	2020-03-06T18:16:39	2
regular	11209	4	2020-03-04T22:44:36	1
node235:				
couchbase-net-sdk/2.6.1.0	11210	406561	2020-03-10T08:21:02	4
couchbase-net-sdk/2.7.0.0	11210	56	2020-03-03T22:04:54	2
proxy	11209	4	2020-03-06T18:16:40	2
regular	11209	5	2020-03-04T22:43:59	1

Bucket configuration overview

Part 1

Bucket	Type	Quota (MB)	Bucket Priority	Eviction Policy	Index Replicas	Conflict Resolution
cdmmp-ai	CB	915	Low	Full	On	Seq
ecdapi	CB	2925	High	Full	On	Seq
Total (2 buckets)	-	3840	-	-	-	-

Part 2

Bucket	Compaction Threshold	Access Control	Flush	Max TTL	Compression Mode	Prod Views/DDocs	Dev Views/DDocs
cdmmp-ai	30%	SASL	On	Off	Passive	12/2	12/2
ecdapi	30%	SASL	On	Off	Passive	22/3	22/3
Total (2 buckets)	-	-	-	-	-	34/5	34/5

Per Data node bucket stats

Bucket	Items	Mem Used (MB)	Metadata%	ARR Active%	ARR Replica%	DWQ
node204:						
cdmmp-ai	0	0	0	100	100	0
ecdapi	3,629,084	2407	0.3	1	1	0
'ecdapi' : % Resident Ratio is too low						
node207:						
cdmmp-ai	0	0	0	100	100	0
ecdapi	3,630,769	2317	0.3	1	1	0
'ecdapi' : % Resident Ratio is too low						
node217:						
cdmmp-ai	0	0	0	100	100	0
ecdapi	3,628,746	2188	0.3	1	1	0
'ecdapi' : % Resident Ratio is too low						
node218:						
cdmmp-ai	0	0	0	100	100	0
ecdapi	3,628,431	2216	0.3	1	1	0
'ecdapi' : % Resident Ratio is too low						
node235:						
cdmmp-ai	0	0	0	100	100	0
ecdapi	3,614,080	2244	0.3	1	1	0
'ecdapi' : % Resident Ratio is too low						

Cluster level totals

Bucket	Total Items	Total Mem Used
cdmmp-ai	0	0
ecdapi	18,131,110	11,372

vBuckets overview

Bucket	Nodes	Total	Online	Replica 1	Replica 2	Replica 3
cdmmp-ai	5	1024	1024	1024	n/a	n/a
ecdapi	5	1024	1024	1024	n/a	n/a

There are no XDCR replications configured on this cluster.

Issues legend

Issue Description	Issue#
OS Version	Issue1
Package-OS Mismatch	Issue2
Installed RAM	Issue3

File ownership	Issue4
Possible Shared Storage	Issue5
MDS Services	Issue6
Memcached Time Jumps	Issue7
Installed CPUs	Issue8
Interface 'ens5' failures	Issue9
Slow operations	Issue10
View Indexing 'ecdapi'	Issue11
View property error	Issue12

Issue severities:

High severity (H) - Issues with this severity are critical in nature and are impacting or are very likely to impact cluster performance and/or stability.
Medium severity (m) - Issues with this severity are causing inefficient use of resources or communication problems or time jumps.

Issues per node matrix

Node	Issue1	Issue2	Issue3	Issue4	Issue5	Issue6
199	H	H	m	m	m	m
204	H	H	m	m	m	-
207	H	H	m	m	m	-
217	H	H	m	m	m	-
218	H	H	m	m	m	-
235	H	H	m	m	m	-
238	H	H	m	m	m	m

Node	Issue7	Issue8	Issue9	Issue10	Issue11	Issue12
199	m	-	-	-	-	-
204	-	m	m	m	m	m
207	-	m	-	-	m	m
217	-	m	-	m	m	m
218	-	m	-	m	m	m
235	-	m	-	m	m	m
238	-	-	m	-	-	-

Node Profiles

OS Information

Node	OS Name	OS Version
node199	Linux 4.15.0-1043-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version
node204	Linux 4.15.0-1021-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version
node207	Linux 4.15.0-1058-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version
node217	Linux 4.15.0-1021-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version
node218	Linux 4.15.0-1021-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version
node235	Linux 4.15.0-1021-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version
node238	Linux 4.15.0-1021-aws	Ubuntu 18.04.1 LTS - Unsupported OS for this version

Uptime and VM Information

Node	Sys Uptime	CB Uptime	Virtual Host

node199	236 days, 18:30:20	-	EC2 - m5.xlarge
node204	397 days, 11:30:41	397 days, 11:32:08	EC2 - r5.large
node207	29 days, 01:28:57	29 days, 01:29:47	EC2 - r5.large
node217	397 days, 11:27:53	397 days, 11:27:11	EC2 - r5.large
node218	397 days, 11:30:43	397 days, 11:31:41	EC2 - r5.large
node235	397 days, 10:02:17	397 days, 10:01:32	EC2 - r5.large
node238	397 days, 09:34:52	-	EC2 - m5.xlarge

MDS and CB Packages

Node	CB Service(s)	CB Server Packages
node199	Index, Query	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server
node204	Data	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server
node207	Data	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server
node217	Data	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server
node218	Data	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server
node235	Data	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server
node238	Index, Query	couchbase-server 6.0.0-1693-1 amd64 Couchbase Server

CPU and Memory

Node	Cores	Total RAM Available	Used RAM (% of Total)	Swap Used (MB)
node199	4	15576 MB	19.0% (2967 / 15576 MB)	-
node204	2	15745 MB	41.1% (6464 / 15745 MB)	-
node207	2	15745 MB	28.9% (4553 / 15745 MB)	-
node217	2	15745 MB	29.2% (4605 / 15745 MB)	-
node218	2	15745 MB	32.3% (5089 / 15745 MB)	-
node235	2	15745 MB	29.9% (4705 / 15745 MB)	-
node238	4	15577 MB	32.5% (5067 / 15577 MB)	-

Couchbase Service Memory Quotas (% of Total RAM Available)

Node	Data	FTS	Index	Analytics	Eventing	Total of Quotas
node199	-	-	13% (2048 MB)	-	-	13% (2048 MB)
node204	25% (3946 MB)	-	-	-	-	25% (3946 MB)
node207	25% (3946 MB)	-	-	-	-	25% (3946 MB)
node217	25% (3946 MB)	-	-	-	-	25% (3946 MB)
node218	25% (3946 MB)	-	-	-	-	25% (3946 MB)
node235	25% (3946 MB)	-	-	-	-	25% (3946 MB)
node238	-	-	13% (2048 MB)	-	-	13% (2048 MB)

Couchbase processes' RSS (in MB)

Node	CouchDB	Memcached	GoXDCR	Projector	Query	Indexer	FTS	Analytics	Eventing
node199	-	-	-	-	140	146	-	-	-
node204	26	2739	7	285	-	-	-	-	-
node207	27	2650	7	149	-	-	-	-	-
node217	27	2537	6	284	-	-	-	-	-
node218	26	2548	6	285	-	-	-	-	-
node235	27	2579	6	286	-	-	-	-	-
node238	-	-	-	-	134	1872	-	-	-

Indexes

Views

Bucket	Directory	Function Name	Function Body

cdmmp-ai	_design/dev_eventdocs	by_airingdatetime	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "event") {
			  if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			  emit(dateToArray(doc.cdapi_document.publishedStartDateTime),doc.cda
			  }
			  }
			  }
			}', 'reduce': '_count

cdmmp-ai	_design/dev_eventdocs	by_source_airingdate	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "event") {
			  if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			  if (typeof doc.cdapi_document.internalIds.sourceId !== "undefined") {
			  var arr1 = [parseInt(doc.cdapi_document.internalIds.sourceId)];
			  var arr2 = dateToArray(doc.cdapi_document.publishedStartDateTime);
			  var emitarr = arr1.concat(arr2);
			  emit(emitarr, doc.cdapi_document.internalIds.programId);
			  }
			  }
			  }
			  }
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_multipartProgramId	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "multipartprogram") {
			  if (typeof doc.cdapi_document.internalIds.multipartProgramId !== "undefined")
			  emit(meta.id, doc.cdapi_document.internalIds.programId);
			  }
			  }
			  }
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_seasonId	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "season") {
			  if (typeof doc.cdapi_document.internalIds.seasonId !== "undefined") {
			  emit(meta.id, doc.cdapi_document.internalIds.seriesId);
			  }
			  }
			  }
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_contributorId	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "contributor") {
			  if (typeof doc.cdapi_document.contributorId !== "undefined") {
			  emit(meta.id, null);
			  }
			  }
			  }
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_sourceId	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "source") {
			  if (typeof doc.cdapi_document.sourceId !== "undefined") {
			  emit(meta.id, null);
			  }
			  }
			  }
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_seriesVariationId	'map': 'function (doc, meta) {
			  if (typeof doc.cdapi_doc_level !== "undefined") {
			  if (doc.cdapi_doc_level.toLowerCase() === "seriesvariation") {
			  if (typeof doc.cdapi_document.internalIds.seriesVariationId !== "undefined") {
			  emit(meta.id, doc.cdapi_document.internalIds.seriesId);
			  }
			  }

			} }', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_programVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "programvariation") {
			if (typeof doc.cdapi_document.internalIds.programVariationId !== "undefined")
			emit(meta.id, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_programId	'map': 'function (doc, meta) {
			emit(meta.id, null);
			}

cdmmp-ai	_design/dev_hierarchylevels	by_seasonVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seasonvariation") {
			if (typeof doc.cdapi_document.internalIds.seasonVariationId !== "undefined")
			emit(meta.id, doc.cdapi_document.internalIds.seasonId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_seriesId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "series") {
			if (typeof doc.cdapi_document.internalIds.seriesId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/dev_hierarchylevels	by_eventId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(meta.id, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/eventdocs	by_airingdatetime	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.publishedStartDateTime),doc.cda
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/eventdocs	by_source_airingdate	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			if (typeof doc.cdapi_document.internalIds.sourceId !== "undefined") {
			var arr1 = [parseInt(doc.cdapi_document.internalIds.sourceId)];
			var arr2 = dateToArray(doc.cdapi_document.publishedStartDateTime);
			var emitarr = arr1.concat(arr2);
			emit(emitarr, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_multipartProgramId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "multipartprogram") {
			if (typeof doc.cdapi_document.internalIds.multipartProgramId !== "undefined")
			emit(meta.id, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_seasonId	'map': 'function (doc, meta) { if (typeof doc.cdapi_doc_level !== "undefined") {

			if (doc.cdapi_doc_level.toLowerCase() === "season") {
			if (typeof doc.cdapi_document.internalIds.seasonId !== "undefined") {
			emit(meta.id, doc.cdapi_document.internalIds.seriesId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_contributorId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "contributor") {
			if (typeof doc.cdapi_document.contributorId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_sourceId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "source") {
			if (typeof doc.cdapi_document.sourceId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_seriesVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seriesvariation") {
			if (typeof doc.cdapi_document.internalIds.seriesVariationId !== "undefined") {
			emit(meta.id, doc.cdapi_document.internalIds.seriesId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_programVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "programvariation") {
			if (typeof doc.cdapi_document.internalIds.programVariationId !== "undefined")
			emit(meta.id, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_programId	'map': 'function (doc, meta) {
			emit(meta.id, null);
			}

cdmmp-ai	_design/hierarchylevels	by_seasonVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seasonvariation") {
			if (typeof doc.cdapi_document.internalIds.seasonVariationId !== "undefined")
			emit(meta.id, doc.cdapi_document.internalIds.seasonId);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_seriesId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "series") {
			if (typeof doc.cdapi_document.internalIds.seriesId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

cdmmp-ai	_design/hierarchylevels	by_eventId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(meta.id, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_multipartProgramId	'map': 'function (doc, meta) { if (typeof doc.cdapi_doc_level !== "undefined") {

			if (doc.cdapi_doc_level.toLowerCase() === "multipartprogram") {
			if (typeof doc.cdapi_document.internalIds.multipartProgramId !== "undefined")
			emit(meta.id, {"programId": doc.cdapi_document.internalIds.programId,
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_seasonId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "season") {
			if (typeof doc.cdapi_document.internalIds.seasonId !== "undefined") {
			emit(meta.id, {"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_contributorId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "contributor") {
			if (typeof doc.cdapi_document.contributorId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_sourceId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "source") {
			if (typeof doc.cdapi_document.sourceId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_seriesVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seriesvariation") {
			if (typeof doc.cdapi_document.internalIds.seriesVariationId !== "undefined") {
			emit(meta.id, {"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_programVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "programvariation") {
			if (typeof doc.cdapi_document.internalIds.programVariationId !== "undefined")
			emit(meta.id, {"programId": doc.cdapi_document.internalIds.programId,
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_programId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "program") {
			if (typeof doc.cdapi_document.internalIds.programId !== "undefined") {
			emit(meta.id, {"seriesId": doc.cdapi_document.internalIds.seriesId, "sea
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_seasonVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seasonvariation") {
			if (typeof doc.cdapi_document.internalIds.seasonVariationId !== "undefined")
			emit(meta.id, {"seasonId": doc.cdapi_document.internalIds.seasonId, "s
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_seriesId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "series") {
			if (typeof doc.cdapi_document.internalIds.seriesId !== "undefined") {
			emit(meta.id, null);

			} } } }', 'reduce': '_count

ecdapi	_design/dev_hierarchylevels	by_eventId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(meta.id, {"programId": doc.cdapi_document.internalIds.programId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_eventdocs	by_airingdatetime	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.publishedStartDateTime), doc.cd
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_eventdocs	by_source_airingdate	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			if (typeof doc.cdapi_document.internalIds.sourceId !== "undefined") {
			var arr1 = [parseInt(doc.cdapi_document.internalIds.sourceId)];
			var arr2 = dateToArray(doc.cdapi_document.publishedStartDateTime);
			var emitarr = arr1.concat(arr2);
			emit(emitarr, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_multipartProgramId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "multipartprogram") {
			if (typeof doc.cdapi_document.internalIds.multipartProgramId !== "undefined")
			emit(meta.id, {"programId": doc.cdapi_document.internalIds.programId,
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_seasonId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "season") {
			if (typeof doc.cdapi_document.internalIds.seasonId !== "undefined") {
			emit(meta.id, {"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_contributorId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "contributor") {
			if (typeof doc.cdapi_document.contributorId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_sourceId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "source") {
			if (typeof doc.cdapi_document.sourceId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_seriesVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seriesvariation") {

			if (typeof doc.cdapi_document.internalIds.seriesVariationId !== "undefined") {
			emit(meta.id, {"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_programVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "programvariation") {
			if (typeof doc.cdapi_document.internalIds.programVariationId !== "undefined")
			emit(meta.id, {"programId": doc.cdapi_document.internalIds.programId,
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_programId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "program") {
			if (typeof doc.cdapi_document.internalIds.programId !== "undefined") {
			emit(meta.id, {"seriesId": doc.cdapi_document.internalIds.seriesId, "sea
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_seasonVariationId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seasonvariation") {
			if (typeof doc.cdapi_document.internalIds.seasonVariationId !== "undefined")
			emit(meta.id, {"seasonId": doc.cdapi_document.internalIds.seasonId, "s
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_seriesId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "series") {
			if (typeof doc.cdapi_document.internalIds.seriesId !== "undefined") {
			emit(meta.id, null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/hierarchylevels	by_eventId	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(meta.id, {"programId": doc.cdapi_document.internalIds.programId}
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	source_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "source") {
			if (typeof doc.cdapi_document.internalIds.sourceId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime), null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	multipartProgram_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "multipartprogram") {
			if (typeof doc.cdapi_document.internalIds.multipartProgramId !== "undefined")
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"programId": doc.cdapi_document.internalIds.programId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	program_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "program") {
			if (typeof doc.cdapi_document.internalIds.programId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),

			{"seriesId": doc.cdapi_document.internalIds.seriesId, "seasonId": } } } }', 'reduce': '_count

ecdapi	_design/dev_history	event_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"sourceId": doc.cdapi_document.internalIds.sourceId, "programId
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	season_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "season") {
			if (typeof doc.cdapi_document.internalIds.seasonId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	programVariation_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "programvariation") {
			if (typeof doc.cdapi_document.internalIds.programVariationId !== "undefined")
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"programId": doc.cdapi_document.internalIds.programId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	seriesVariation_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seriesvariation") {
			if (typeof doc.cdapi_document.internalIds.seriesVariationId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	contributor_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "contributor") {
			if (typeof doc.cdapi_document.internalIds.contributorId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime), null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	series_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "series") {
			if (typeof doc.cdapi_document.internalIds.seriesId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime), null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/dev_history	seasonVariation_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seasonvariation") {
			if (typeof doc.cdapi_document.internalIds.seasonVariationId !== "undefined")
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seasonId": doc.cdapi_document.internalIds.seasonId,
			"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/eventdocs	by_airingdatetime	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.publishedStartDateTime), doc.cd
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/eventdocs	by_source_airingdate	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			if (typeof doc.cdapi_document.internalIds.sourceId !== "undefined") {
			var arr1 = [parseInt(doc.cdapi_document.internalIds.sourceId)];
			var arr2 = dateToArray(doc.cdapi_document.publishedStartDateTime);
			var emitarr = arr1.concat(arr2);
			emit(emitarr, doc.cdapi_document.internalIds.programId);
			}
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	source_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "source") {
			if (typeof doc.cdapi_document.internalIds.sourceId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime), null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	multipartProgram_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "multipartprogram") {
			if (typeof doc.cdapi_document.internalIds.multipartProgramId !== "undefined")
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"programId": doc.cdapi_document.internalIds.programId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	program_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "program") {
			if (typeof doc.cdapi_document.internalIds.programId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seriesId": doc.cdapi_document.internalIds.seriesId, "seasonId":
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	event_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "event") {
			if (typeof doc.cdapi_document.internalIds.eventId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"sourceId": doc.cdapi_document.internalIds.sourceId, "programIc
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	season_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "season") {
			if (typeof doc.cdapi_document.internalIds.seasonId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	programVariation_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "programvariation") {

			if (typeof doc.cdapi_document.internalIds.programVariationId !== "undefined")
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"programId": doc.cdapi_document.internalIds.programId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	seriesVariation_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seriesvariation") {
			if (typeof doc.cdapi_document.internalIds.seriesVariationId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	contributor_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "contributor") {
			if (typeof doc.cdapi_document.internalIds.contributorId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime), null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	series_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "series") {
			if (typeof doc.cdapi_document.internalIds.seriesId !== "undefined") {
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime), null);
			}
			}
			}
			}', 'reduce': '_count

ecdapi	_design/history	seasonVariation_lastModified	'map': 'function (doc, meta) {
			if (typeof doc.cdapi_doc_level !== "undefined") {
			if (doc.cdapi_doc_level.toLowerCase() === "seasonvariation") {
			if (typeof doc.cdapi_document.internalIds.seasonVariationId !== "undefined")
			emit(dateToArray(doc.cdapi_document.lastModifiedDateTime),
			{"seasonId": doc.cdapi_document.internalIds.seasonId,
			"seriesId": doc.cdapi_document.internalIds.seriesId});
			}
			}
			}
			}', 'reduce': '_count

Global Secondary Indexes

Indexes Summary

Bucket	Index Name	Node	Status	Progress	Type
cdmmp-ai	jobid-details-index	node238	Ready	100	plasma
cdmmp-ai	cdmmpai-primary-index	node199	Ready	100	plasma
cdmmp-ai	metaid-index	node238	Ready	100	plasma
cdmmp-ai	cbkey-statustoken-index	node199	Ready	100	plasma
ecdapi	ecdapi-primary-index	node238	Ready	100	plasma

Found 5 indexes. All are missing replicas

Index Definitions

Index Name	Index Definition
cbkey-statustoken-index	CREATE INDEX `cbkey-statustoken-index` ON `cdmmp-ai`(`cbKey`,`statusToken`) WHERE (`aspect` is not missing)
cdmmpai-primary-index	CREATE PRIMARY INDEX `cdmmpai-primary-index` ON `cdmmp-ai`

ecdapi-primary-index	CREATE PRIMARY INDEX `ecdapi-primary-index` ON `ecdapi`
jobid-details-index	CREATE INDEX `jobid-details-index` ON `cdmmp-ai`((`details`.`jobId`)) WHERE ((`aspect` is not missing) and (`cbKey` is not missing))
metaid-index	CREATE INDEX `metaid-index` ON `cdmmp-ai`((meta().`id`))