

AU 2021 - NoSQL Basics & Fundamentals Assignment

Divyansh khatri

1. Import `beer-sample` bucket.

Solution.

We can use the Couchbase web interface itself to import the beer-sample bucket.

Steps.

- Go to Settings.
- Go to Sample Buckets.
- Choose beer-sample from Available Samples list.
- Click on Load Sample Data

The screenshot shows the Couchbase Settings page for Cluster1. The left sidebar has 'Settings' selected. The main content area is titled 'Sample Buckets'. It contains two sections: 'Available Samples' (listing 'gamesim-sample') and 'Installed Samples' (listing 'beer-sample' and 'travel-sample'). A blue button labeled 'Load Sample Data' is visible. The top navigation bar shows 'localhost' and the user 'Divyansh'.

2. Write a join query to fetch the Top 10 brewery(type="beer") and their country(type="brewery") which produces more varieties of beers.

Solution.

1. We first have to create two Datasets first namely,

- beers

2. CREATE DATASET beers ON `beer-sample` WHERE `type` = "beer";

- breweries

CREATE DATASET breweries ON `beer-sample` WHERE `type` = "brewery";

2.1 The following command can be used to get the top 10 breweries and the number of types of beer they produce.

```
SELECT META(breweries).id, COUNT(*) from breweries
INNER JOIN beers ON META(breweries).id = beers.brewery_id
GROUP BY META(breweries).id
ORDER BY COUNT(*) DESC
LIMIT 10;
```

The screenshot shows the Cluster1 Analytics interface. On the left, there's a sidebar with various navigation options like Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The 'Analytics' option is selected. In the main area, there's an 'Analytics Query Editor' with the following SQL query:

```
1 SELECT META(breweries).id, COUNT(*) from breweries
2 INNER JOIN beers ON META(breweries).id = beers.brewery_id
3 GROUP BY META(breweries).id
```

Below the editor are 'Execute' and 'Explain' buttons, and a status message: '✓ success | elapsed: 299.68ms | execution: 274.74ms | docs scanned: 7303 | docs returned: 10 | size: 395 bytes'. To the right of the editor is a 'Datasets' section listing 'beer-data' with datasets 'beers' and 'breweries'. Further down is a 'Couchbase Buckets' section listing 'Beer', 'beer-sample', and 'travel-sample'. The 'Analytics Query Results' section displays the query results as JSON, showing 10 brewery entries with their IDs and counts:

```
1 [ 
2   { 
3     "id": "midnight_sun_brewing_co",
4     "$1": 57
5   },
6   { 
7     "id": "rogue_ales",
8     "$1": 49
9   },
10  { 
11    "id": "anheuser_busch",
12    "$1": 38
13  },
14  { 
15    "id": "troegs_brewing",
16    "$1": 37
17  },
18  { 
19    "id": "egan_brewing",
20    "$1": 37
21  },
22  { 
23    "id": "boston_beer_company",
24    "$1": 36
25  },
26  { 
27    "id": "titletown_brewing",
28    "$1": 34
29  }
30 ]
```

A 'Refresh' button is located at the bottom right of the results table.

2.2 The following command can be used to get the details of the top 10 breweries and all the types of beers with the countries that they produce.

```
SELECT * from breweries
INNER JOIN beers ON META(breweries).id = beers.brewery_id
GROUP BY META(breweries).id
ORDER BY COUNT(*) DESC
LIMIT 10;
```

The screenshot shows the Cluster1 Analytics interface. On the left, there's a sidebar with various navigation options like Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The 'Analytics' option is selected. The main area has tabs for 'Workbench' and 'Monitor'. In the 'Workbench' tab, there's an 'Analytics Query Editor' containing the following SQL-like query:

```

1 SELECT * from breweries
2 INNER JOIN beers ON META(breweries).id = beers.brewery_id
3 GROUP BY META(breweries).id
4 ORDER BY COUNT(*) DESC
5 LIMIT 10;

```

Below the editor, there are 'Execute' and 'Explain' buttons, and a status message: '✓ success | elapsed: 292.24ms execution: 272.93ms docs scanned: 7303 docs returned: 10 size: 537110 bytes'. To the right of the editor is an 'Analytics Query Results' panel showing a JSON response for one brewery. The results are paginated with '1' at the top and '19' at the bottom. The JSON output is as follows:

```

1 [
2   {
3     "id": "midnight_sun_brewing_co",
4     "s1": [
5       {
6         "breweries": {
7           "name": "Midnight Sun Brewing Co.",
8           "city": "Anchorage",
9           "state": "Alaska",
10          "code": "99507",
11          "country": "United States",
12          "phone": "1-907-344-1179",
13          "website": "http://www.midnightsunbrewing.com/",
14          "type": "brewery",
15          "updated": "2010-07-22 20:00:20",
16          "description": "Since firing up its brew kettle in 1995, Midnight Sun Brewing Company has become a serious yet creative force on the American brewing front. From concept to glass, Midnight Sun relies on an art marries science approach, mixing tradition with innovation, to design and craft bold, distinctive beers for Alaska...and beyond. We at Midnight Sun find inspiration in the untamed spirit and rugged beauty of the Last Frontier and develop unique beers with equally appealing names and labels. But the company's true focus remains in its dedication to producing consistently high-quality beers that provide satisfying refreshment in all seasons... for Alaskans and visitors alike. From our Pacific Northwest locale, we offer our wonderful beers on draft throughout Alaska and in 22 -ounce bottles throughout Alaska and Oregon. We invite you to visit our hardworking, little brewery in South Anchorage every chance you get!",
17          "address": [
18            "8111 Dimond Hook Drive"
19          ],
20        }
21      ]
22    }
23  ]

```

On the far right, there are sections for 'Datasets' (Dataverse: Default, beers, breweries) and 'Couchbase Buckets' (Beer, beer-sample, travel-sample). There's also a 'Refresh' button.

3. Write a mapreduce to get the number of breweries based on country.
Please attach the mapreduce code and json output screenshot.

Solution.

The map query for the following query will be -

MAP QUERY

```
function (doc, meta) {
  if(doc.type == "brewery"){
    emit(doc.country, doc.name);
  }
}
```

REDUCE QUERY

_count

1.1. Output for Full Cluster Data Set

The screenshot shows the Apache CouchDB Futon interface on a web browser. The URL is `localhost`. The left sidebar has tabs for **Indexes**, **Search**, **Analytics**, **Eventing**, and **Views**. The **Views** tab is selected. In the main area, there is a **View Index Code** section with the following code:

```
function (doc, meta) {
  if(doc.type == "brewery"){
    emit(doc.country, doc.name);
  }
}
```

Below this is a **Map** section with the code above. To the right is a **Reduce (built in: _count, _sum, _stats)** section containing the code `1 _count`. There are **Make Copy** and **Save Changes** buttons.

At the bottom, there is a **Results** section with a dropdown filter set to `?limit=6&stale=false&connection_timeout=60000&inclusive_end=true&skip=0&full_set=true&group=true`. A **Development Time Subset** dropdown is set to **Full Cluster Data Set**. There are navigation buttons (**<**, **>**) and a **Show Results** button. The results table has columns **Key** and **Value**.

Key	Value
"" undefined	1
"Argentina" undefined	2
"Aruba" undefined	1
"Australia" undefined	14

1.2. Output json for the Full Cluster Data Set which shows the country name and the number of breweries each country has.

The screenshot shows a web browser window with the URL `localhost:8092/beer-newsample/_design/dev_beer/_view/beer-sample?limit=6&stale=false&connection_timeout=60000&inclusive_end=true&skip=0&full_set=true&group=true`. The page displays the following JSON output:

```
{"rows": [{"key": "", "value": 1}, {"key": "Argentina", "value": 2}, {"key": "Aruba", "value": 1}, {"key": "Australia", "value": 14}, {"key": "Austria", "value": 10}, {"key": "Belgium", "value": 99} ]}
```

2.1 Output for Development Time Subset

The screenshot shows the Couchbase Web Console interface. At the top, there are tabs for 'Assignment' and 'Cluster1 - Enterprise Edition 7.0.0 build...'. Below the tabs, there are sections for 'Analytics', 'Eventing', and 'Views'. The 'Views' section is active, showing a 'Map' function and a 'Reduce' function.

```
Map
1 function (doc, meta) {
2   if(doc.type == "brewery"){
3     emit(doc.country, doc.name);
4   }
5 }
```

```
Reduce (built in: _count, _sum, _stats)
1 _count
```

Below the functions, the results are displayed under the heading 'Development Time Subset' and 'Full Cluster Data Set'. The results table has columns 'Key' and 'Value'.

Key	Value
"Australia" undefined	1
"Belgium" undefined	1
"Germany" undefined	1
"Japan" undefined	1
"Namibia" undefined	1
"United States" undefined	12

2.2 Output json for the Development Time Subset which shows the country name and the number of breweries each country has.

The screenshot shows the Couchbase Web Console interface. At the top, there are tabs for 'Assignment' and 'Cluster1 - Enterprise Edition 7.0.0 build...'. Below the tabs, there are sections for 'Analytics', 'Eventing', and 'Views'. The 'Views' section is active, showing a 'Map' function and a 'Reduce' function.

```
Map
1 function (doc, meta) {
2   if(doc.type == "brewery"){
3     emit(doc.country, doc.name);
4   }
5 }
```

```
Reduce (built in: _count, _sum, _stats)
1 _count
```

Below the functions, the results are displayed under the heading 'Development Time Subset' and 'Full Cluster Data Set'. The results table has columns 'Key' and 'Value'.

Key	Value
"Australia" undefined	1
"Belgium" undefined	1
"Germany" undefined	1
"Japan" undefined	1
"Namibia" undefined	1
"United States" undefined	12

At the bottom of the page, the raw JSON output is shown:

```
{"rows": [{"key": "Australia", "value": 1}, {"key": "Belgium", "value": 1}, {"key": "Germany", "value": 1}, {"key": "Japan", "value": 1}, {"key": "Namibia", "value": 1}, {"key": "United States", "value": 12} ]}
```

4. XDCR:

- a) Add a new bucket “Brewery”.

We can create a new bucket by going to buckets and then clicking on ADD BUCKET on the top right corner.

The screenshot shows the Cluster1 interface with the 'Buckets' tab selected. A modal dialog box titled 'Add Data Bucket' is open. In the 'Name' field, 'Brewery' is entered. Under 'Memory Quota' (in megabytes per server node), '8457 MB' is specified. Below this, a progress bar indicates disk usage: 'other buckets (400 MB)' (blue), 'this bucket (8.25 GB)' (orange), and 'remaining (0 B)' (grey). The 'Bucket Type' section shows 'Couchbase' selected. At the bottom of the dialog are 'Cancel' and 'Add Bucket' buttons.

The screenshot shows the Cluster1 interface with the 'Buckets' tab selected. The list of buckets now includes 'beer-sample', 'Brewery', and 'travel-sample'. The 'Brewery' bucket has been added with the following details: 1,412 items, 100% resident, 47 ops/sec, 4.6MB / 8.25GB RAM used/quota, and 1.88MB disk used. The other two buckets ('beer-sample' and 'travel-sample') also have their respective statistics listed.

Bucket	Items	resident	ops/sec	RAM used/quota	disk used	Documents	Scopes & Collections
beer-sample	7,303	100%	0	8.22MB / 200MB	5.34MB	Documents	Scopes & Collections
Brewery	1,412	100%	47	4.6MB / 8.25GB	1.88MB	Documents	Scopes & Collections
travel-sample	63,186	100%	0	50.8MB / 200MB	43.1MB	Documents	Scopes & Collections

- b) Create a XDCR with a filter(type='brewery') to replicate only the brewery entity from `beer-sample` bucket.

Solution. To replicate the beer-sample bucket using XDCR, we will follow the following steps-

- Click on XDCR and select ADD REPLICATION on the top right corner.
- Select Filter Replication from the options and type the filter expression -

REGEXP_CONTAINS(type, "brewery")

The screenshot shows the 'XDCR Add Replication' configuration page. The 'Replicate From Bucket' dropdown is set to 'beer-sample'. The 'Remote Cluster' dropdown is set to 'Cluster1'. The 'Remote Bucket' input field is set to 'Brewery'. A note below states: 'The bucket beer-sample will be replicated to Cluster1. Missing targets on the remote will be ignored and that data will not be replicated.' Under 'Mapping Rules', there are three toggle switches: 'Specify scopes, collections, and mapping' (off), 'Migrate collections' (off), and 'Filter replication' (on). The 'Filter Expression' field contains 'REGEXP_CONTAINS(type, "brewery")'. Below it is a 'Test Filter' button and a 'Delete' field. Under 'Deletion Filters', there are three checkboxes: 'Do not replicate document expirations' (off), 'Do not replicate DELETE operations' (off), and 'Remove TTL from replicated items' (off). At the bottom are 'Save Replication' and 'Cancel/Reset' buttons.

- Click on Save replication.

The screenshot shows the 'XDCR Replications' page. It lists a single outgoing replication entry. The 'source bucket' is 'beer-sample' with a 'filter' applied. The 'destination bucket' is 'Brewery'. The 'remote cluster' is 'Cluster1'. The 'status' is 'starting up ...'. On the right side of the row are 'Delete', 'Edit', and 'Pausing' buttons. The left sidebar shows the navigation menu with 'XDCR' selected.

This is the result of replication into the Brewery bucket.

The screenshot shows the Cluster1 web interface at the URL `localhost`. The title bar includes tabs for "Assignment" and "Meet - ijn-brcn-xjn". The top right corner shows "Cluster1 - Enterprise Edition 7.0.0 build 3739", "activity", "help", and a user "Divyansh". The main header is "Cluster1 > Buckets" with a "ADD BUCKET" button. On the left, a sidebar menu lists various sections: Dashboard, Servers, **Buckets**, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The "Buckets" section is currently selected. A search bar labeled "filter buckets..." is present. The main content area displays a table of buckets:

name	items	resident	ops/sec	RAM used/quota	disk used	Documents	Scopes & Collections
beer-sample	7,303	100%	0	8.22MB / 200MB	5.34MB	Documents	Scopes & Collections
Brewery	1,412	100%	47	4.6MB / 8.25GB	1.88MB	Documents	Scopes & Collections
travel-sample	63,186	100%	0	50.8MB / 200MB	43.1MB	Documents	Scopes & Collections

5. CLI:

- Add a new bucket “Beer”.

We will use this command to create a bucket named beer -

```
./couchbase-cli bucket-create -c couchbase://127.0.0.1 --username Divyansh \ --password Divyansh1 --bucket Beer --bucket-type couchbase \ --bucket-ramsize 1024
```

The terminal window shows the command being run and its output:

```
(base) divyanshkhatri@Divyanshs-MacBook-Pro bin % ./couchbase-cli bucket-create -c couchbase://127.0.0.1 --username Divyansh \
--password Divyansh1 --bucket Beer --bucket-type couchbase \
--bucket-ramsize 1024
SUCCESS: Bucket created
(base) divyanshkhatri@Divyanshs-MacBook-Pro bin %
```

The screenshot shows the Couchbase Cluster1 > Buckets interface. On the left, a sidebar lists various navigation options: Dashboard, Servers, Buckets (selected), Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The main content area displays the 'Beer' bucket details. The 'Beer' bucket has 1,329 items, is 100% resident, and has 0 ops/sec. It uses 4.96MB / 1GB of RAM and 4.21MB of disk space. The 'beer-sample' and 'travel-sample' buckets also have their respective details shown. A legend indicates cluster quota (8.64 GB) for memory and total cluster storage (460 GB) for disk. Buttons for 'Drop', 'Compact', and 'Edit' are visible at the bottom right.

b) Using CLI - do a cbexport of the entire `beer-sample`

We can use the following command for cbexport-

```
./cbexport json -c couchbase://127.0.0.1 -u Divyansh -p Divyansh1 -b beer-sample -o /Users/divyanshkhatri/Desktop/beer-sample-cbexport.json -f lines -t 4 --scope-field couchbaseScope --collection-field couchbaseCollection
```

```
(base) divyanshkhatri@Divyanshs-MacBook-Pro bin % ./cbexport json -c couchbase://127.0.0.1 -u Divyansh -p Divyansh1 -b beer-sample -o /Users/divyanshkhatri/Desktop/beer-sample-cbexport.json -f lines -t 4 --scope-field couchbaseScope --collection-field couchbaseCollection
JSON exported to '/Users/divyanshkhatri/Desktop/beer-sample-cbexport.json' successfully
Documents exported: 7303 Documents skipped: 0
(base) divyanshkhatri@Divyanshs-MacBook-Pro bin %
```

```

1 {"abv":0,"brewery_id":"dempsey_s_restaurant_brewery","category":"North American Ale","couchbaseCollection":"_default","couchbaseScope":"_default","description":"","ibu":0,"name":"Red Rooster","srm":0,"style":"American-Style Pale Ale","type":"beer","upc":0,"updated":"2010-07-22 20:00:20"}
2 {"abv":0,"brewery_id":"mia_and_pia_s_pizzeria_and_brewhouse","category":"North American Ale","couchbaseCollection":"_default","couchbaseScope":"_default","description":"","ibu":0,"name":"Kalamath Basin IPA","srm":0,"style":"American-Style India Pale Ale","type":"beer","upc":0,"updated":"2010-07-22 20:00:20"}
3 {"abv":0,"brewery_id":"yakima_brewing_and_malting_grant_s_ales","couchbaseCollection":"_default","couchbaseScope":"_default","description":"","ibu":0,"name":"Spiced Ale","srm":0,"type":"beer","upc":0,"updated":"2010-07-22 20:00:20"}
4 {"abv":3.6,"brewery_id":"sullivan_s_black_forest_brew_haus_grill","category":"North American Ale","couchbaseCollection":"_default","couchbaseScope":"_default","description":"Our take on the classic light golden ale, with crisp refreshing flavors.","ibu":0,"name":"Woody's Light","srm":0,"style":"American-Style Pale Ale","type":"beer","upc":0,"updated":"2010-07-22 20:00:20"}
5 {"abv":0,"brewery_id":"black_creek_historic_brewery","category":"Other Style","couchbaseCollection":"_default","couchbaseScope":"_default","description":"Porter is a dark-coloured beer developed in the 1700s. Porter has a heavier flavour and aroma and a slightly sweet taste. Its name probably originates in the belief that this strong, nourishing drink was ideal for hard-working porters and labourers. Black Creek Porter is wonderful on its own, or with salty snacks.\n\n","ibu":0,"name":"Black Creek Porter","srm":0,"style":"Smoke Beer","type":"beer","upc":0,"updated":"2011-06-08 07:10:06"}
6 {"abv":11.1,"brewery_id":"egan_brewing","category":"British Ale","couchbaseCollection":"_default","couchbaseScope":"_default","description":"","ibu":0,"name":"Cow Palace Scotch Ale 2001","srm":0,"style":"Scotch Ale","type":"beer","upc":0,"updated":"2010-07-22 20:00:20"}
7 {"abv":0,"brewery_id":"mercury_brewing_company","category":"North American Ale","couchbaseCollection":"_default","couchbaseScope":"_default","description":"","ibu":0,"name":"1084 Barleywine","srm":0,"style":"American-Style Barley Wine Ale","type":"beer","upc":0,"updated":"2010-07-22 20:00:20"}

```

c) And do a cbimport with “brewery_id” as primary key. As a result, in the new bucket - only “beer” documents will be imported with their respective brewery name as meta().id
Solution.

We can use the following command for cbimport-

```
./cbimport json -c couchbase://127.0.0.1 -u Divyansh -p Divyansh1 -b Beer -f lines -d file://Users/divyanshkhatri/Desktop/beer-sample-cbexport.json -t 4 -g %brewery_id%
```

