

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 a brief description: 'Sample buckets contain example data, views, and indexes for your experimentation.' Below this, it says 'Sample buckets — like all buckets in Couchbase Server 5.0+ — can only be accessed by a user with privileges for that bucket.' There are 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 at the bottom of the list.

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.

2.1 The following command can be used to produce the top 10 breweries, there country as well as the different types of beers they produce.

```
SELECT a.brewery_id,
       b.country,
       COUNT(*)
FROM `beer-sample` a
      JOIN `beer-sample` b ON a.brewery_id = META(b).id
      AND a.type="beer"
```

```

GROUP BY a.brewery_id,
        b.country
ORDER BY COUNT(a.brewery_id) DESC
LIMIT 10;

```

The screenshot shows the Apache Nifi Cluster1 Query interface. The left sidebar has 'Query' selected. The main area has a 'Query Editor' tab open with the following SQL code:

```

1 SELECT a.brewery_id,
2        b.country,
3        COUNT(*)
4  FROM `beer-sample` a
5    JOIN `beer-sample` b ON a.brewery_id = META(b).id
6   AND a.type="beer"
7 GROUP BY a.brewery_id,
8        b.country
9 ORDER BY COUNT(a.brewery_id) DESC
10 LIMIT 10;

```

Below the editor are three buttons: 'Execute', 'Explain', and 'Advise'. The 'Execute' button is highlighted. To its right, status information is displayed: success, 4 min ago, elapsed: 210.5ms, execution: 210.4ms, docs: 10, size: 1025 bytes. To the right of the status is a 'Data Insights' panel. The 'beer-sample' section shows one collection with 7303 documents, where 19.3% have 'type' = "brewery". The 'travel-sample' section shows six collections with various document counts for categories like airline, airport, hotel, landmark, and route.

Query Results

\$1	brewery_id	country
57	midnight_sun_brewing_co	United States
49	rogue_ales	United States
38	anheuser_busch	United States
37	egan_brewing	United States
37	troegs_brewing	United States
36	boston_beer_company	United States
34	titeltown_brewing	United States

This screenshot is identical to the one above, showing the same query execution and results. The 'beer-sample' dataset shows 7303 documents with 19.3% having 'type' = "brewery". The 'travel-sample' dataset shows 6 collections with various document counts for categories like airline, airport, hotel, landmark, and route.

Query Results

\$1	brewery_id	country
57	midnight_sun_brewing_co	United States
49	rogue_ales	United States
38	anheuser_busch	United States
37	egan_brewing	United States
37	troegs_brewing	United States
36	boston_beer_company	United States
34	titeltown_brewing	United States
34	f_x_matt_brewing	United States
33	sierra_nevada_brewing_co	United States
32	stone_brewing_co	United States

2. The second method involves analytics querying. First have to create two Datasets first namely,

- beers

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

- breweries

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

2.2The 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 main area has tabs for Workbench, Monitor, and Analytics Query Editor. The Analytics Query Editor tab is active, displaying 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
4 ORDER BY COUNT(*) DESC
5 LIMIT 10;
```

Below the query, it says "Execute" and "Explain". The status bar indicates "success" and "elapsed: 299.68ms | execution: 274.74ms | docs scanned: 7303 | docs returned: 10 | size: 395 bytes". To the right, there's a "Datasets" panel showing "beer-data" with datasets "beers" and "breweries", and a "Couchbase Buckets" panel showing buckets "Beer", "beer-sample", and "travel-sample". The "Analytics Query Results" panel on the right shows the JSON output of the query, which is a list of 10 objects, each representing a brewery with its ID and the count of beer types it produces. The first few results are:

```
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": "titlertown_brewing",
28    "$1": 34
}
```

2.3 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 Query Editor interface. On the left, a sidebar lists 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'. The 'Workbench' tab is active, showing the 'Analytics Query Editor' with the following SQL 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 query, there are buttons for 'Execute' (highlighted in blue), 'Explain', and 'History (49/49)'. The status bar indicates success with 'elapsed: 292.24ms execution: 272.93ms docs scanned: 7303 docs returned: 10 size: 537110 bytes'. To the right, there are sections for 'Datasets' (Dataverse: Default, beers, breweries) and 'Couchbase Buckets' (Beer, beer-sample, travel-sample). The 'Analytics Query Results' section displays a JSON document for a brewery, with the 'JSON' tab selected. 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  ]

```

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

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 **Reduce (built in: _count, _sum, _stats)** section with the code:

```
_count
```

Under the **Results** section, there is a filter dropdown set to `?limit=6&stale=false&connection_timeout=60000&inclusive_end=true&skip=0&full_set=true&group=true`. The time subset is set to **Full Cluster Data Set**. 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' section displays the output of the development time subset. It includes a filter: ?limit=6&stale=false&connection_timeout=60000&inclusive_end=true&skip=0&full_set=&group=true. The results table has columns 'Key' and 'Value'.

Key	Value
"Australia"	1
"Belgium"	1
"Germany"	1
"Japan"	1
"Namibia"	1
"United States"	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' section displays the raw JSON output. It includes a filter: ?limit=6&stale=false&connection_timeout=60000&inclusive_end=true&skip=0&full_set=&group=true. The output is a list of objects:

```
{"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 title 'Cluster1 > Buckets'. On the left, there is a sidebar with various navigation options like Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The 'Buckets' option is selected. A modal window titled 'Add Data Bucket' is open in the center. In the 'Name' field, 'Brewery' is typed. Under 'Memory Quota' (in megabytes per server node), '8457 MB' is entered. Below this, there is a progress bar showing disk usage: 'other buckets (400 MB)', 'this bucket (8.25 GB)', and 'remaining (0 B)'. Under 'Bucket Type', the 'Couchbase' radio button is selected. At the bottom of the modal are 'Cancel' and 'Add Bucket' buttons. The background shows other buckets listed: 'beer-sample' and 'travel-sample'. Disk usage statistics are also visible at the top right: '5.07MB' and '43.1MB'.

The screenshot shows the Cluster1 interface with the title 'Cluster1 > Buckets'. The sidebar on the left is identical to the previous screenshot. The main area displays a table of buckets. The table has columns: name, items, resident, ops/sec, RAM used/quota, and disk used. The data is as follows:

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

- 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 'id of document to test...' input 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). A 'Save Replication' button is at the bottom.

- 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, there are three buttons: 'Delete', 'Edit', and 'Pausing'. 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
```

```
(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 Server management interface at the URL 127.0.0.1. The top navigation bar includes links for various dashboard sections like Header butt..., Spring @Qu..., flexbox - Sp..., The Movie..., Merge two..., Cluster1 - E..., Start Page, Cluster1 - E..., Assignment, activity, help, and Divyansh. A red banner at the top states "BETA · PREVIEW MODE · UNSUPPORTED · NOT FOR USE IN PRODUCTION". The main title is "Cluster1 > Buckets". On the left, a sidebar menu lists categories such as Dashboard, Servers, Buckets (which is selected), Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The central area displays a table of buckets:

name	items	resident	ops/sec	RAM used/quota	disk used	
Beer	0	100%	0	3.86MB / 8.25GB	273KB	Documents Scopes & Collections
beer-sample	7,303	100%	0	7.07MB / 200MB	4.75MB	Documents Scopes & Collections
travel-sample	63,186	100%	0	50.8MB / 200MB	43.3MB	Documents Scopes & Collections

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 1750s. 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"}

```

The command has successfully exported all the documents from the beer-sample into the beer-sample-cbexport.json file. The above shows some of the documents present inside the json file.

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%
```

