

Final Assignment - Querying data in NoSQL databases



Estimated time needed: **90** minutes.

It is highly recommended that you finish the [Setup and Practice Assignment Lab](#) before you proceed with this Assignment.

About This SN Labs Cloud IDE

This Skills Network Labs Cloud IDE provides a hands-on environment for course and project related labs. It utilizes Theia, an open-source IDE (Integrated Development Environment) platform, that can be run on desktop or on the cloud. To complete this lab, we will be using the Cloud IDE based on Theia and Cassandra and MongoDB running in a Docker container. You will also need an instance of Cloudant running in IBM Cloud.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session, to avoid losing your data.

Scenario

You are a data engineer at a data analytics consulting company. Your company prides itself in being able to efficiently handle data in any format on any database on any platform. Analysts in your office need to work with data on different databases, and data in different formats. While they are good at analyzing data, they count on you to be able to move data from external sources into various databases, move data from one type of database to another, and be able to run basic queries on various databases.

Objectives

In this assignment you will:

- replicate a Cloudant database.
- create indexes on a Cloudant database.
- query data in a Cloudant database.
- import data into a MongoDB database.
- query data in a MongoDB database.
- export data from MongoDB.
- import data into a Cassandra database.
- query data in a Cassandra database.

Note - Screenshots

Throughout this lab you will be prompted to take screenshots and save them on your own device. These screenshots will be needed to be uploaded for peer review in the next section of the course. You can use various free screengrabbing tools to do this or use your operating system's shortcut keys to do this (for example Alt+PrintScreen in Windows).

Exercise 1 - Check the lab environment

Before you proceed with the assignment :

- Check if you have the 'couchimport' tool installed on the lab, otherwise install it.
- Check if you have the 'mongoimport' and 'mongoexport' tools installed on the lab, otherwise install them.
- Check if the environment variable CLOUDANTURL is set, otherwise set it.

Exercise 2 - Working with a Cloudant database

Task 1 - Replicate a remote database into your Cloudant instance

Using the replication page in the Cloudant dashboard, replicate the below source database using one time replication,

Source

Type : Remote Database

Database URL: <https://examples.cloudant.com/query-movies>

Authentication : None

into a local database named movies.

Once the replication is done take a screenshot of the database's dashboard (which shows database names, number of rows etc,).

Name the screenshot as **1-replication.jpg**. (images can be saved with either .jpg or .png extension)

Task 2 - Create an index for the "director" key, on the 'movies' database using the HTTP API

Take a screenshot of the command you used and the output.

Name the screenshot as **2-index-director.jpg**. (images can be saved with either .jpg or .png extension)

Task 3 - Write a query to find all movies directed by 'Richard Gage' using the HTTP API

Take a screenshot of the command you used and the output.

Name the screenshot as **3-query-director.jpg**. (images can be saved with either .jpg or .png extension)

Task 4 - Create an index for the "title" key, on the 'movies' database using the HTTP API

Take a screenshot of the command you used and the output.

Name the screenshot as **4-index-title.jpg**. (images can be saved with either .jpg or .png extension)

Task 5 - Write a query to list only the "year" and "director" keys for the 'Top Dog' movies using the HTTP API

Take a screenshot of the command you used and the output.

Name the screenshot as **5-query-title.jpg**. (images can be saved with either .jpg or .png extension)

Task 6 - Export the data from the 'movies' database into a file named 'movies.json'

Take a screenshot of the command you used and the output.

Name the screenshot as **6-couchexport.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 3 - Working with a MongoDB database

Task 7 - Import 'movies.json' into mongodb server into a database named 'entertainment' and a collection named 'movies'

Take a screenshot of the command you used and the output.

Name the screenshot as **7-mongoimport.jpg**. (images can be saved with either .jpg or .png extension)

Task 8 - Write a mongodb query to find the year in which most number of movies were released

► [Click here for Hint](#)

Take a screenshot of the command you used and the output.

Name the screenshot as **8-mongo-query.jpg**. (images can be saved with either .jpg or .png extension)

Task 9 - Write a mongodb query to find the count of movies released after the year 1999

Take a screenshot of the command you used and the output.

Name the screenshot as **9-mongo-query.jpg**. (images can be saved with either .jpg or .png extension)

Task 10. Write a query to find out the average votes for movies released in 2007

► [Click here for Hint](#)

Take a screenshot of the command you used and the output.

Name the screenshot as **10-mongo-query.jpg**. (images can be saved with either .jpg or .png extension)

Task 11 - Export the fields **_id**, "title", "year", "rating" and "director" from the 'movies' collection into a file named **partial_data.csv**

Take a screenshot of the command you used and the output.

Name the screenshot as **11-mongoexport.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 4 - Working with a Cassandra database

Task 12 - Import 'partial_data.csv' into cassandra server into a keyspace named 'entertainment' and a table named 'movies'

Note: While creating the table **movies** make the all the columns as text columns including the id column.

Take a screenshot of the command you used and the output.

Name the screenshot as **12-cassandra-import.jpg**. (images can be saved with either .jpg or .png extension)

Task 13 - Write a cql query to count the number of rows in the 'movies' table

Take a screenshot of the command you used and the output.

Name the screenshot as **13-cassandra-query.jpg**. (images can be saved with either .jpg or .png extension)

Task 14 - Create an index for the "rating" column in the 'movies' table using cql

Take a screenshot of the command you used and the output.

Name the screenshot as 14-cassandra-index.jpg. (images can be saved with either .jpg or .png extension)

Task 15 - Write a cql query to count the number of movies that are rated 'G'.

Take a screenshot of the command you used and the output.

Name the screenshot as 15-cassandra-query.jpg. (images can be saved with either .jpg or .png extension)

End of assignment.

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja




Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2021-06-08	0.4	Malika Singla	Updated task 10 to match rubric
2021-04-23	0.3	Steve Ryan	Review pass
2021-04-16	0.2	Ramesh Sannareddy	Reorganised the Exercises and Tasks

2021-04-14	0.1	Ramesh Sannareddy	Created initial version
------------	-----	-------------------	-------------------------

Copyright (c) 2021 IBM Corporation. All rights reserved.

1 – REPLICATION

Databases					Database name	Create Database	{ } JSON		
Your Databases									
Name	Size	# of Docs	Partitioned	Actions					
_replicator	7.9 KB	3	No						
movies	3.1 MB	9411	No						
training	51 bytes	1	No						
training_replica	51 bytes	1	No						

2- INDEX DIRECTOR

```
theia@theiadocker-joaocosta1:/home/project$ export CLOUDANTURL="https://apikey-v2-ty2o15b10
5tqvr9z3eruq0b0qo2l9u1m9qyedadx9y:9e2fdeae7984e6f4745a9562e7728c92@ecfb0a1c-d6af-4142-bd37
-12bbd45287d0-bluemix.cloudantnosqldb.appdomain.cloud"
theia@theiadocker-joaocosta1:/home/project$ curl $CLOUDANTURL
{"couchdb":"Welcome","version":"3.2.1","vendor":{"name":"IBM Cloudant","version":"8243","va
riant":"paas"},"features":["geo","search","access-ready","iam","partitioned","pluggable-sto
rage-engines","scheduler"],"features_flags":["partitioned"]}
theia@theiadocker-joaocosta1:/home/project$ curl $CLOUDANTURL/_all_dbs
["_replicator","movies","training","training_replica"]
theia@theiadocker-joaocosta1:/home/project$ curl -X POST $CLOUDANTURL/movies/_index \
> -H"Content-Type: application/json" \
> -d'{
> "index": {
> "fields": ["director"]
> }
> }'
{"result":"created","id":"_design/e654aff2816dc36cee9c1c00041203ac74282421","name":"e654aff
2816dc36cee9c1c00041203ac74282421"}
theia@theiadocker-joaocosta1:/home/project$
```

3- QUERY DIRECTOR

```
theia@theiadocker-joaocosta1:/home/project$ curl -X POST $CLOUDANTURL/movies/_find \
> -H"Content-Type: application/json" \
> -d'{ "selector":
> {
> "director": "Richard Gage"
> }
> }'
{"docs": [
{"_id": "2628664", "_rev": "1-4f5f4002310ff1a6d808e1f5363f62e7", "title": "9/11: Explosive Evidence - Experts Speak Out", "year": 2012, "rating": "Unrated", "runtime": "90 min", "genre": ["Documentary", "Crime", "History"], "director": "Richard Gage", "writer": ["Richard Gage"], "cast": ["Steven Barash", "Mark Basile", "Steven Dusterwald", "Jeffrey Farrer"], "poster": "http://ia.media-imdb.com/images/M/MV5BMTUxNDc1MDc2Nl5BMl5BanBnXkFtZTgwNDI3NzA2MDE@._V1_SX300.jpg", "imdb": {"rating": 8.6, "votes": 94, "id": "tt2628664"}}
],
"bookmark": "g1AAABJeJzLYWBgYMpgSmHgKy5JLCrJTq2MT81PzkzJBYPqzG5kZWZiZmYCK0WDSOUAJRpAsT1BmckZiUVqCe2J6a1YWAK_DE2I"}

theia@theiadocker-joaocosta1:/home/project$
```

4- INDEX TITLE

```
theia@theiadocker-joaocosta1:/home/project$ curl -X POST $CLOUDANTURL/movies/_index \
> -H"Content-Type: application/json" \
> -d'{
> "index": {
> "fields": ["title"]
> }
> }'
{"result": "created", "id": "_design/794d3a1331c7f3304fb7715d2e2b21095f9e1be9", "name": "794d3a1331c7f3304fb7715d2e2b21095f9e1be9"}

theia@theiadocker-joaocosta1:/home/project$
```

5-QUERY TITLE

```
theia@theiadocker-joaocosta1:/home/project$ curl -X POST $CLOUDANTURL/movies/_find \
> -H"Content-Type: application/json" \
> -d'{
>   "selector":
>   {
>     "title":"Top Dog"
>   },
>   "fields": [
>     "year",
>     "director"
>   ]}'
{"docs":[{"year":1978,"director":"Feliks Falk"}
],
"bookmark": "g1AAAABCeJzLYWBgYMPgSmHgKy5JLCrJTq2MT8lPzkzJBYqzmluYGpiDJDlGkj1AYUaQHtIfGCS3
56VhYAMt0REA"}

theia@theiadocker-joaocosta1:/home/project$
```

6- COUCH EXPORT

```
theia@theiadocker-joaocosta1:/home/project$ couchexport --url $CLOUDANTURL --db movies --ty
pe jsonl > movies.json
couchexport Output 500 [500] +0ms
couchexport Output 500 [1000] +743ms
couchexport Output 500 [1500] +709ms
couchexport Output 500 [2000] +709ms
couchexport Output 500 [2500] +751ms
couchexport Output 500 [3000] +718ms
couchexport Output 500 [3500] +691ms
couchexport Output 500 [4000] +724ms
couchexport Output 500 [4500] +706ms
couchexport Output 500 [5000] +717ms
couchexport Output 500 [5500] +710ms
couchexport Output 500 [6000] +700ms
couchexport Output 500 [6500] +700ms
couchexport Output 500 [7000] +691ms
couchexport Output 500 [7500] +703ms
couchexport Output 500 [8000] +720ms
couchexport Output 500 [8500] +740ms
couchexport Output 500 [9000] +701ms
couchexport Output 414 [9414] +634ms
theia@theiadocker-joaocosta1:/home/project$
```

7- MONGO IMPORT

```
theia@theiadocker-joaocosta1:/home/project$ mongoimport -u root -p MzE40TEtam9hb2Nv --authenticationD
atabase admin --db entertainment --collection movies --file movies.json
2022-01-31T10:16:59.230+0000 connected to: mongodb://localhost/
2022-01-31T10:17:01.027+0000 9413 document(s) imported successfully. 0 document(s) failed to impor
t.
```

8- MONGO QUERY

```
> db.movies.aggregate([
... {
... "$group":{
... "_id":"$year",
... "moviecount":{"$sum":1}
... }
... },
... {
... "$sort":{"moviecount":-1}
... },
... {
... "limit":1}
... ]
```

10- MONGO QUERY

```
> db.movies.aggregate([
... {
... "$match":{
... "year":2007
... }
... },
... {"$group":{
... "_id":"$year",
... "average":{"$avg":$imdb.votes"}
... ]}
```

11- MONGO EXPORT

```
> mongoexport -u root -p MzE40TEtam9hb2Nv --authenticationDatabase admin --db training --collection movies --out partial_data.csv --type=csv --fields _id,title,year,rating,director
```

12- CASSANDRA IMPORT

```
cassandra@cqlsh> CREATE TABLE movies(
... "_id" text,
... cast text,
... director text,
... genre text,
... imdb text
... )
... COPY entertainment.movies(_id,cast,director,genre,imdb) FROM 'partial_data.csv' WITH DELIMITER=';' AND HEADER=TRUE;
```


13- CASSANDRA QUERY

```
cassandra@cqlsh> use entertainment  
... SELECT COUNT(*) FROM movies
```

14 CASSANDRA INDEX

```
cassandra@cqlsh> use entertainment  
... CREATE INDEX rating_index ON entertainment.movies (rating)
```

15 – CASSANDRA QUERY

```
... SELECT COUNT (*) FROM movies WHERE rating = "G";
```