# Week5_FinalAssignment

**Instructions**
Now that you are equipped with the knowledge and skills to work with several different NoSQL databases you have the opportunity in the final project to practice and apply your skills by working with different databases to move and analyze 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 the offices need to work with data on different databases, and with 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.

**Grading Criteria**
There are a total of 20 points possible for this final project.

Your final assignment will be graded by your peers who are also completing this assignment within the same session. Your grade will be based on the following tasks:

**Task 1**: Replicate a remote database into your Cloudant instance. (1 pts)

**Task 2**: Create an index for key "director", on the database movies using the HTTP API. (1pts)

```
curl -X POST $SERVERURL/movies/_index \
-H"Content-Type: application/json" \
-d'{
    "index": {
        "fields": ["director"]
    }
}'
```

where `SERVERURL` = `"https://username:password@host"`

**Task 3**: Write a query to find all movies directed by Richard Gage using the HTTP API. (1 pts)

```
curl -X POST $SERVERURL/movies/_find \
-H "Content-Type: application/json" \
-d '{"selector":{"Director":"Richard Gage"}}'
```

**Task 4**: Create an index for key "title", on the database movies using the HTTP API. (1 pts)

```
curl -X POST $SERVERURL/movies/_index \
-H"Content-Type: application/json" \
-d'{
    "index": {
        "fields": ["title"]
    }
}'
```

**Task 5:** Write a query to list only the keys year and director for the movie `Top Dog` using the HTTP API. (2 pts)

```
curl -X POST $SERVERURL/movies/_find \
-H "Content-Type: application/json" \
-d '{"selector":{"title":"Top Dog"},
"fields": ["year","Director"]}'
```

**Task 6:** Export the data from movies database into a file named movies.json. (2 pts)

`couchexport --url $CLOUDANTURL --db movies --type jsonl > movies.json`

**Task 7:** Import movies.json into mongodb server into a database named entertainment and collection named movies. (1 pts)

```
mongoimport -u root -p MTU5NDEtam9hb3Bh --authenticationDatabase admin --db
entertainment --collection movies --file movies.json
```

**Task 8:** Write a mongodb query to find the year in which most number of movies were released. (2 pts)

```

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

**Task 9:** Write a mongodb query to find the count of movies released after the year 1999. (1 pts)

```
db.movies.find({year: {$gt: 1999}}).count()
```

**Task 10**. Write a query to find out the average votes for movies released in 2007. (2 pts)

```
db.movies.aggregate( [
{ $match : { year : 2007 }
},
{
$group: {

_id:null,
avg: { $avg: "$Votes"}
}

}


])
```

**Task 11** - Export the fields _id, title, year, rating and director from movies collection into a file named partial_data.csv. (2 pts)

```

mongoexport -u root -p MTU5NDEtam9hb3Bh --authenticationDatabase admin --db
entertainment --collection movies --out partial_data.csv --type=csv --fields
_id,title,year,rating,Director
```

**Task 12** - Import partial_data.csv into cassandra server into a keyspace named entertainment and table named movies. (1 pts)

```
CREATE KEYSPACE entertainment
WITH replication = {'class':'SimpleStrategy', 'replication_factor' : 3};

use entertainment;
CREATE TABLE movies(
id text PRIMARY KEY,
title text,
year int,
rating text,
Director text
);

COPY entertainment.movies(id,title,year,rating, Director) FROM
'partial_data.csv' WITH DELIMITER=',' AND HEADER=TRUE;
```

**Task 13** - Write a cql query to count the number of rows in the movies table. (1 pts)

```
SELECT COUNT(*) FROM movies;
```

**Task 14** - Create an index for the column rating in the movies table using cql. (1 pts)

`create index rating_index on movies(rating);`

**Task 15** - Write a cql query to count the number of in the movies that are rated 'G'. (1 pts)

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