Module 12 Challenge

Start Assignment

Due Thursday by 11:59pm **Points** 100 **Submitting** a text entry box or a website url

Before You Begin

1. Create a new repository for this project called `nosql-challenge`. **Do not add this homework to an existing repository**.
2. Clone the new repository to your computer.
3. Add your Jupyter notebook starter files and your Resources folder containing `establishments.json` to this folder.
4. Push the changes to GitHub.

Files

Download the following files to help you get started:

Module 12 Challenge files (https://static.bc-edx.com/data/dl-1-2/m12/lms/starter/Starter_Code.zip)

Instructions

The UK Food Standards Agency evaluates various establishments across the United Kingdom, and gives them a food hygiene rating. You've been contracted by the editors of a food magazine, *Eat Safe, Love*, to evaluate some of the ratings data in order to help their journalists and food critics decide where to focus future articles.

Part 1: Database and Jupyter Notebook Set Up

Use `NoSQL_setup_starter.ipynb` for this section of the challenge.

1. Import the data provided in the `establishments.json` file from your Terminal. Name the database `uk_food` and the collection `establishments`. Copy the text you used to import your data from your Terminal to a markdown cell in your notebook.
2. Within your notebook, import the libraries you need: PyMongo and Pretty Print (`pprint`).
3. Create an instance of the Mongo Client.
4. Confirm that you created the database and loaded the data properly:
   - List the databases you have in MongoDB. Confirm that `uk_food` is listed.
   - List the collection(s) in the database to ensure that `establishments` is there.
   - Find and display one document in the `establishments` collection using `find_one` and display with `pprint`.
5. Assign the `establishments` collection to a variable to prepare the collection for use.

Part 2: Update the Database

Use `NoSQL_setup_starter.ipynb` for this section of the challenge.

The magazine editors have some requested modifications for the database before you can perform any queries or analysis for them. Make the following changes to the `establishments` collection:

1. An exciting new halal restaurant just opened in Greenwich, but hasn't been rated yet. The magazine has asked you to include it in your analysis. Add the following information to the database:

```
{
    "BusinessName":"Penang Flavours",
    "BusinessType":"Restaurant/Cafe/Canteen",
    "BusinessTypeID":"",
    "AddressLine1":"Penang Flavours",
    "AddressLine2":"146A Plumstead Rd",
    "AddressLine3":"London",
    "AddressLine4":"",
    "PostCode":"SE18 7DY",
    "Phone":"",
    "LocalAuthorityCode":"511",
    "LocalAuthorityName":"Greenwich",
    "LocalAuthorityWebSite":"http://www.royalgreenwich.gov.uk",
    "LocalAuthorityEmailAddress":"health@royalgreenwich.gov.uk",
    "scores":{
        "Hygiene":"",
        "Structural":"",
        "ConfidenceInManagement":""
    },
    "SchemeType":"FHRS",
    "geocode":{
        "longitude":"0.08384000",
        "latitude":"51.49014200"
    },
    "RightToReply":"",
    "Distance":4623.9723280747176,
    "NewRatingPending":True
}
```

2. Find the BusinessTypeID for "Restaurant/Cafe/Canteen" and return only the `BusinessTypeID` and `BusinessType` fields.

3. Update the new restaurant with the `BusinessTypeID` you found.

4. The magazine is not interested in any establishments in Dover, so check how many documents contain the Dover Local Authority. Then, remove any establishments within the Dover Local Authority from the database, and check the number of documents to ensure they were deleted.

5. Some of the number values are stored as strings, when they should be stored as numbers.
    1. Use `update_many` to convert `latitude` and `longitude` to decimal numbers.
    2. Use `update_many` to convert `RatingValue` to integer numbers.

Part 3: Exploratory Analysis

Eat Safe, Love has specific questions they want you to answer, which will help them find the locations they wish to visit and avoid.

Use `NoSQL_analysis_starter.ipynb` for this section of the challenge.

Some notes to be aware of while you are exploring the dataset:

- `RatingValue` refers to the overall rating decided by the Food Authority and ranges from 1-5. The higher the value, the better the rating.
    - **Note:** This field also includes non-numeric values such as 'Pass', where 'Pass' means that the establishment passed their inspection but isn't given a number rating. We will coerce non-numeric values to nulls during the database setup before converting ratings to integers.
- The scores for Hygiene, Structural, and ConfidenceInManagement work in reverse. This means, the higher the value, the worse the establishment is in these areas.

Use the following questions to explore the database, and find the answers, so you can provide them to the magazine editors.

Unless otherwise stated, for each question:

- Use `count_documents` to display the number of documents contained in the result.
- Display the first document in the results using `pprint`.
- Convert the result to a Pandas DataFrame, print the number of rows in the DataFrame, and display the first 10 rows.

1. Which establishments have a hygiene score equal to 20?
2. Which establishments in London have a `RatingValue` greater than or equal to 4?

   Hint: The London Local Authority has a longer name than "London" so you will need to use `$regex` as part of your search.
3. What are the top 5 establishments with a `RatingValue` of 5, sorted by lowest hygiene score, nearest to the new restaurant added, "Penang Flavours"?

   Hint: You will need to compare the geocode to find the nearest locations. Search within 0.01 degree on either side of the latitude and longitude.
4. How many establishments in each Local Authority area have a hygiene score of 0? Sort the results from highest to lowest, and print out the top ten local authority areas.

   Hint: You will need to use the `aggregation` method to answer this.

   The first 5 rows of your resulting DataFrame should look something like this:

	_id	count
0	Thanet	1130

	_id	count
1	Greenwich	882
2	Maidstone	713
3	Newham	711
4	Swale	686

Requirements

Part 1: Database and Jupyter Notebook Set Up (15 points)

To receive all points, your Jupyter notebook setup file must have all of the following:

- Include the `mongoimport` command text you used to import `establishments.json` in a markdown cell at the beginning of your Jupyter notebook file (3 points)
- The `mongoimport` command text correctly drops any existing `establishments` collection before importing `establishments.json` into MongoDB (2 points)
- The database is named `uk_food` and the collection is named `establishments` (2 points)
- Correctly imports PyMongo and Pretty Print (2 points)
- An instance of the Mongo Client is created (1 point)
- Lists the databases you have in Mongo, which includes `uk_food` (1 point)
- Lists the collection(s) in the `uk_food` database, which includes `establishments` in the output (1 point)
- Uses `find_one()` and `pprint` to display one document in the `establishments` collection (2 points)
- The `establishments` collection is assigned to a variable (1 point)

Part 2: Update the Database (20 points)

To receive all points, your Jupyter notebook setup file must have all of the following:

- The supplied data for the "Penang Flavours" restaurant is correctly inserted into the `establishments` collection (3 points)
- A query is performed to find the `BusinessTypeID` for "Restaurant/Cafe/Canteen" and returns only the `BusinessTypeID` and `BusinessType` fields (3 points)
- The "Penang Flavours" document is updated with the correct value for `BusinessTypeID` (3 points)
- A query is correctly performed to delete all the documents in the collection where "Dover Local Authority" is the value for `LocalAuthorityName` (3 points)
- A `count_documents()` check is performed before and after the removal of the Dover documents to ensure the documents were removed (4 points)

- An `update_many()` query is performed to convert the `latitude` and `longitude` fields from strings to decimal numbers and `RatingValue` to integers (4 points)

Part 3: Exploratory Analysis (55 points)

To receive all points, your Jupyter notebook analysis file must have all of the following:

Question 1: Which establishments have a hygiene score equal to 20? (8 points)

- A query is correctly performed to find the establishments with a hygiene score of 20 (2 points)
- `count_documents()` is used to list the correct number of documents (answer: 41) (2 points)
- The first result is printed using `pprint` (2 points)
- The results are converted to a Pandas DataFrame and displays the first 10 rows (2 points)

Question 2: Which establishments in London have a `RatingValue` greater than or equal to 4? (12 points)

- A query is correctly performed to find the establishments in London with a `RatingValue` greater than or equal to 4 (4 points)
- The query uses the `$regex` operator to locate the London establishments (2 points)
- `count_documents()` is used to list the correct number of documents (answer: 33) (2 points)
- The first result is printed using `pprint` (2 points)
- The results are converted to a Pandas DataFrame and displays the first 10 rows (2 points)

Question 3: What are the top 5 establishments with a `RatingValue` of 5, sorted by lowest hygiene score, nearest to the new restaurant added, "Penang Flavours"? (15 points)

- A query is correctly performed to find the establishments within 0.01 degree of the "Penang Flavours" restaurant (4 points)
- The query also limits the results to establishments with a `RatingValue` of 5 (2 points)
- The query uses the `sort()` method in PyMongo to sort in ascending order on the hygiene score (2 points)
- The query uses the `limit()` method in PyMongo to limit the results to 5 (2 points)
- All five results are printed using `pprint` (3 points)
- The results are converted to a Pandas DataFrame and displayed (2 points)

Question 4: How many establishments in each Local Authority area have a hygiene score of 0? Sort the results from highest to lowest, and print out the top ten local authority areas. (20 points)

- An aggregation pipeline is built to include a match query, group, and sort (3 points)
- The match query matches documents with a hygiene score of 0 (2 points)
- The group step of the pipeline is grouped on `LocalAuthorityName` and counts the number of documents (4 points)
- The sort step of the pipeline sorts the count of the documents in descending order (2 points)

- The aggregation pipeline is correctly sent to the `aggregate()` method (2 points)
- The results from the aggregation query is cast as a list and then saved to a variable (2 points)
- The first ten results are printed using `pprint` (3 points)
- The results are converted to a Pandas DataFrame and displays the first 10 rows (2 points)

Deployment and Submission (6 points)

To receive all points, you must:

- Submit a link to a GitHub repository that's cloned to your local machine and contains your files (2 points)
- Use the command line to add your files to the repository (2 points)
- Include appropriate commit messages in your files (2 points)

Comments (4 points)

To receive all points, your code must:

- Be well commented with concise, relevant notes that other developers can understand (4 points)

Grading

This assignment will be evaluated against the requirements and assigned a grade according to the following table:

Grade	Points
A (+/-)	90+
B (+/-)	80–89
C (+/-)	70–79
D (+/-)	60–69
F (+/-)	< 60

Submission

As a reminder, the deliverables for this Challenge are as follows:

- Deliverable 1: A Jupyter notebook containing code that imports the data and sets up and updates the uk_food database.
- Deliverable 2: A Jupyter notebook containing code that performs the exploratory analysis queries in the database.