# Lab 3

**Question 1: Open MongoDB Atlas and connect to the restaurant collection that was used in Week 12 and 13. Filter the collection with {restaurant\_id: "99999996"} condition. There will be zero (0) document found; take a screenshot and lace it inside the DOCX file.**

A screenshot of a computer

Description automatically generated

**Question 2: Explain why no document was inserted or created even though HTTP PUT request was made inside the DOCX file. (Hint: Understand the code in step 5 of Exercise 1 to the fullest and explain why).**

No document was created because the HTTP PUT request only adds a message to the Redis queue using messageQueue.add().

**Question 3: Run node read\_queue\_update\_mongo.js on another command line tool window (Terminal on MacOS, Cygwin on Windows). Take a screenshot of the output and place it inside the DOCX file.**

A screen shot of a computer

Description automatically generated

**Question 4: Open MongoDB Atlas and connect to the restaurant collection that was used in Week 12 and 13. Filter the collection with {restaurant\_id: "99999996"} condition. There will be one (1) document found; take a screenshot and lace it inside the DOCX file.**

A screenshot of a computer

Description automatically generated

**Question 6: Open MongoDB Atlas and connect to the restaurant collection that was used in Week 12 and 13. Filter the collection with {restaurant\_id: "99999996"} condition. Take a screenshot and place it inside the DOCX file; also, explain what changed.**A screenshot of a computer

Description automatically generated

The value of newRate in the document with restaurant\_id: "99999996" changed to "C"

**Question 7: If your script is correctly implemented, a document whose restaurant\_id is 99999996 would’ve been deleted. Copy and paste the updated script, read\_queue\_update\_mongo.js, into the DOCX file.**

import { MongoClient } from 'mongodb';

import Queue from 'bull';

import { } from 'dotenv/config';

const uri = process.env.MONGODB\_URI;

const client = new MongoClient(uri);

async function run() {

  const messageQueue = new Queue('messageQueue');

  messageQueue.process(async (job) => {

    console.log('Processing : ' + job.id);

    const database = client.db('sample\_restaurants');

    const restaurants = database.collection('restaurants');

    const query = { restaurant\_id: job.data.restaurant\_id };

    console.log('job.data.updates: ', job.data.updates);

    if (job.data.updates === "\"delete\"") {

      await restaurants.deleteOne(query);

      console.log(`Deleted restaurant\_id: ${job.data.restaurant\_id}`);

    }

    else {

      await restaurants.updateOne(query,

        { $set: JSON.parse(job.data.updates) },

        { upsert: true }

      );

      console.log('Processed: ' + JSON.stringify(job.data));

    }

    return;

  });

}

run().catch(console.dir);