```
var AWS = require("aws-sdk");
const express = require("express")
const app = express()
const port = 3000

AWS.config.update({
    region: "us-east-1",
    endpoint: "http://dynamodb.us-east-1.amazonaws.com/",
  });

  var dynamodb = new AWS.DynamoDB();

  const bucket = {
      Bucket: "csu44000assign2useast20",
      Key: "moviedata.json"
  }
  app.listen(port, ()=>console.log(`Example app listening on port ${port}!`))

  app.get('/', (_, res) => {
    res.sendFile(__dirname + '/index.html');                //send to index.html
  })

  app.get('/check', async (_, res) => {                     //check if table exists
    var params = {
        TableName: "Movies"
    };
    dynamodb.describeTable(params, function(err, data) {
        if (err) return res.json({ success: false });       //if false table doesnt
exist
        else     res.json({ success: true });               //if true table exists
    })
  })

  app.get('/delete', async (_, res) => {                    //delete table
    var params = {
        TableName : "Movies"
    };

    dynamodb.deleteTable(params, function(err, data) {
        if (err) {
            console.error("Unable to delete table. Error JSON:", JSON.stringify(err,
null, 2));
            return res.json({ success: false });
        } else {
            console.log("Deleted table. Table description JSON:",
JSON.stringify(data, null, 2));
            return res.json({ success: true });
        }
    });

})

  app.get('/create', async (_, res) => {
    try {
        const s3 = new AWS.S3({endpoint: "https://s3.us-east-1.amazonaws.com"})
//Amazon Simple Storage Service
        let response = await s3.getObject(bucket).promise()
//get bucket
        let data = JSON.parse(response.Body.toString('utf-8'))
 //get data reable
```

```
55          allMovies = data
     //set allMovies = data
56      }
57      catch(err){
58          console.log(err);
59      }
60      var params = {
61          TableName : "Movies",
62          KeySchema: [
63              { AttributeName: "year", KeyType: "HASH"},  //Partition key
64              { AttributeName: "title", KeyType: "RANGE" }  //Sort key
65          ],
66          AttributeDefinitions: [
67              { AttributeName: "year", AttributeType: "N" },
68              { AttributeName: "title", AttributeType: "S" }
69          ],
70          ProvisionedThroughput: {
71              ReadCapacityUnits: 5,
72              WriteCapacityUnits: 5
73          }
74      };
75
76      dynamodb.createTable(params, function(err, data) {
77          if (err) {
78              console.error("Unable to create table. Error JSON:", JSON.stringify(err,
    null, 2));
79          } else {
80              console.log("Created table. Table description JSON:",
    JSON.stringify(data, null, 2));
81          }
82      });
83      var docClient = new AWS.DynamoDB.DocumentClient();
84
85      var params = {
86          TableName: 'Movies' /* required */
87        };
88        dynamodb.waitFor('tableExists', params, function(err, data) {
89          if (err) {console.log(err, err.stack);} // an error occurred
90          else  {
91              allMovies.forEach(function(movie) {
92                  var params = {
93                      TableName: "Movies",
94                      Item: {
95                          "year":  movie.year,
96                          "title": movie.title,
97                          "rank":  movie.info.rank
98                      }
99                  };
100
101             docClient.put(params, function(err, data) {
102                if (err) {
103                    console.error("Unable to add movie", movie.title, ". Error JSON:",
    JSON.stringify(err, null, 2));
104                } else {
105                    console.log("PutItem succeeded:", movie.title);
106                }
107             });
108         });
109         }
110       });
```

```
      return res.json({ success: true });
})

    app.get('/query/:title/:year', async (req, res) => {
        var docClient = new AWS.DynamoDB.DocumentClient();
        let year = parseInt(req.params.year)
        let title = req.params.title
        var params = {
            TableName : "Movies",
            KeyConditionExpression: "#yr = :yyyy and begins_with(title, :letter1)",
            ExpressionAttributeNames:{
                "#yr": "year"
            },
            ExpressionAttributeValues: {
                ":yyyy": year,
                ":letter1": title
            }
        };

        docClient.query(params, function(err, data) {
            if (err) {
                console.log("Unable to query. Error:", JSON.stringify(err, null, 2));
            } else {
                console.log("Query succeeded.");
                let result = []
                data.Items.forEach(function(item) {
                    let movie = {year: item.year, title: item.title, rank: item.rank}
                    result.push(movie)
                });
                return res.json({ result: result });
            }
        });
    })
```