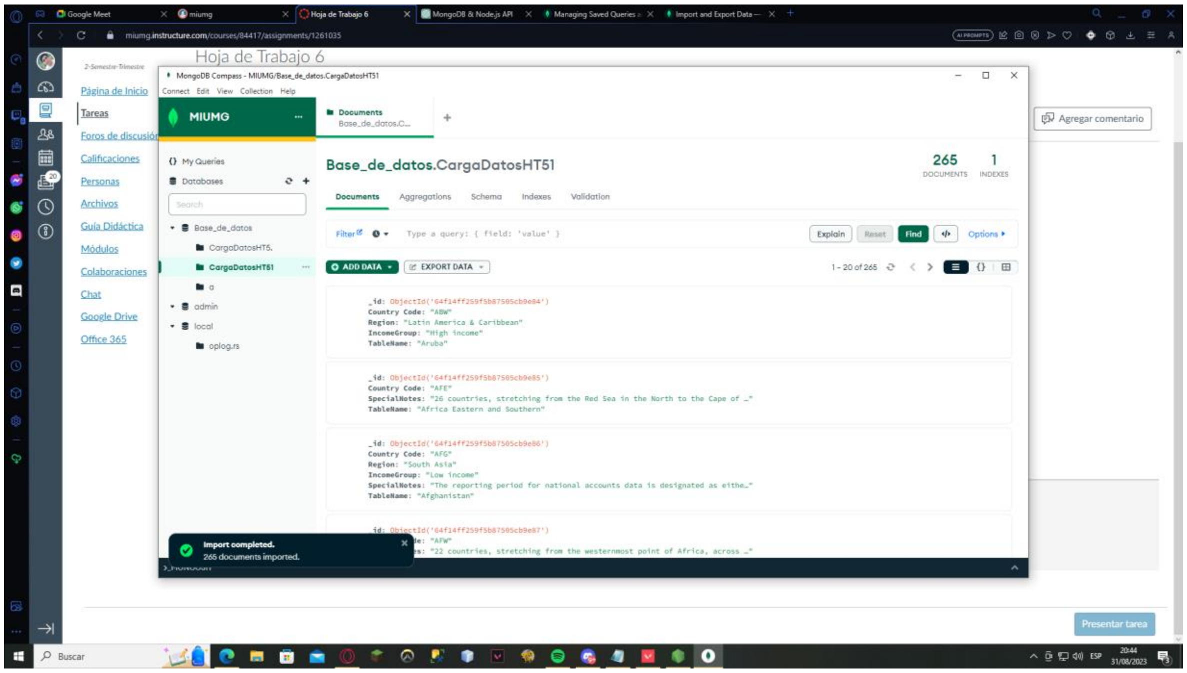
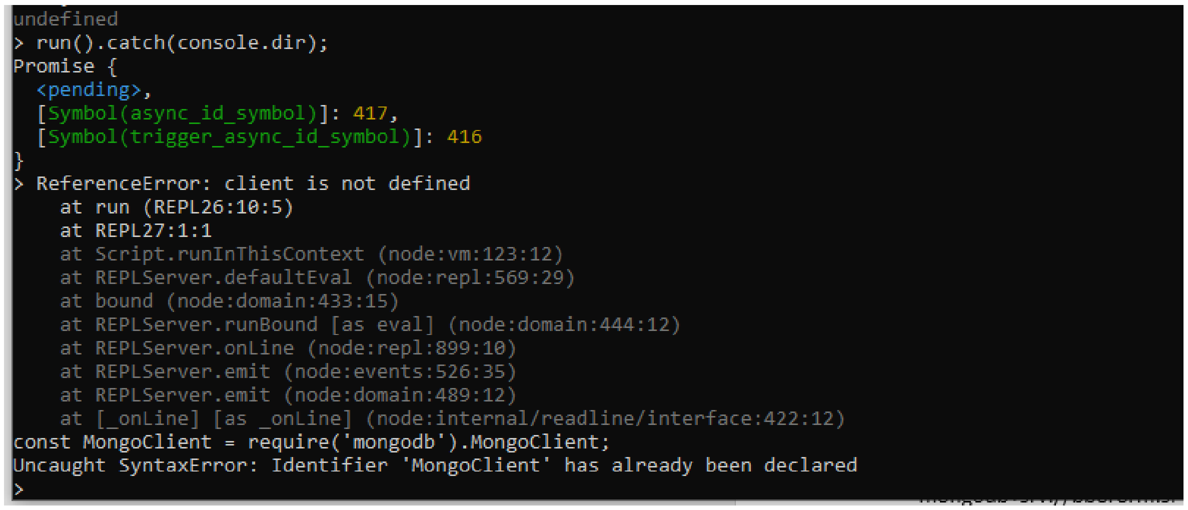


```
_id: ObjectId('64f14ff259f5b87505cb9e84')
Country Code: "ABW"
Region: "Latin America & Caribbean"
IncomeGroup: "High income"
TableName: "Aruba"

_id: ObjectId('64f14ff259f5b87505cb9e85')
Country Code: "AFE"
SpecialNotes: "26 countries, stretching from the Red Sea in the North to the Cape of …"
TableName: "Africa Eastern and Southern"

_id: ObjectId('64f14ff259f5b87505cb9e86')
Country Code: "AFG"
Region: "South Asia"
IncomeGroup: "Low income"
SpecialNotes: "The reporting period for national accounts data is designated as eithe…"
TableName: "Afghanistan"

_id: ObjectId('64f14ff259f5b87505cb9e87')
Country Code: "AFW"
SpecialNotes: "22 countries, stretching from the westernmost point of Africa, across …"
```

Conexión de nodejs a mongodb

```
const MongoClient = require('mongodb').MongoClient;



const url =
'mongodb+srv://bbororm:sPlaIjAEupA0n6LC@cluster0.5i66oz0.mongodb.net/?retryWrites=true&
w=majority';



MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, (err, client) => {
  if (err) {
    console.error('Error al conectar a la base de datos:', err);
    return;
  }
  console.log('Conexión exitosa a la base de datos');

});
```



```
undefined
> run().catch(console.dir);
Promise {
  <pending>,
  [Symbol(async_id_symbol)]: 417,
  [Symbol(trigger_async_id_symbol)]: 416
}
> ReferenceError: client is not defined
    at run (REPL26:10:5)
    at REPL27:1:1
    at Script.runInThisContext (node:vm:123:12)
    at REPLServer.defaultEval (node:repl:569:29)
    at bound (node:domain:433:15)
    at REPLServer.runBound [as eval] (node:domain:444:12)
    at REPLServer.onLine (node:repl:899:10)
    at REPLServer.emit (node:events:526:35)
    at REPLServer.emit (node:domain:489:12)
    at [_onLine] [as _onLine] (node:internal/readline/interface:422:12)
const MongoClient = require('mongodb').MongoClient;
Uncaught SyntaxError: Identifier 'MongoClient' has already been declared
>
```

API de nodejs

```
const express = require('express');

const MongoClient = require('mongodb').MongoClient;

const ObjectID = require('mongodb').ObjectID;


const app = express();

const port = 3000;


const url = 'mongodb://localhost:27017/Base_de_datos';

app.use(express.json());


app.use((req, res, next) => {

  MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, (err, client) => {

    if (err) {

      console.error('Error al conectar a la base de datos:', err);

      return;

    }

    const db = client.db();
```

```
        req.db = db;
        next();
    });
});

app.get('/records', (req, res) => {
    const collection = req.db.collection('CargaDatosHT51'); // Cambia según tus detalles

    collection.find({}).toArray((err, records) => {
        if (err) {
            console.error('Error al consultar la colección:', err);
            res.status(500).send('Error de servidor');
            return;
        }
        res.json(records);
    });
});

// Obtener un registro por su ID
app.get('/records/:id', (req, res) => {
    const collection = req.db.collection('CargaDatosHT51');

    const recordId = req.params.id;

    collection.findOne({ _id: new ObjectID(recordId) }, (err, record) => {
        if (err) {
            console.error('Error al consultar el registro:', err);
            res.status(500).send('Error de servidor');
            return;
        }
        if (!record) {
            res.status(404).send('Registro no encontrado');
            return;
        }
        res.json(record);
    });
});

// Crear un nuevo registro
app.post('/records', (req, res) => {
    const collection = req.db.collection('CargaDatosHT51');

    const newRecord = req.body;

    collection.insertOne(newRecord, (err, result) => {
        if (err) {
            console.error('Error al insertar el registro:', err);
            res.status(500).send('Error de servidor');
            return;
        }
        res.json(result.ops[0]);
    });
});

// Actualizar un registro
app.put('/records/:id', (req, res) => {
    const collection = req.db.collection('CargaDatosHT51');
```

```
  const recordId = req.params.id;
  const updatedData = req.body;

  collection.findOneAndUpdate(
    { _id: new ObjectID(recordId) },
    { $set: updatedData },
    { returnOriginal: false },
    (err, result) => {
      if (err) {
        console.error('Error al actualizar el registro:', err);
        res.status(500).send('Error de servidor');
        return;
      }
      res.json(result.value);
    }
  );
});

// Eliminar un registro
app.delete('/records/:id', (req, res) => {
  const collection = req.db.collection('CargaDatosHT51');

  const recordId = req.params.id;

  collection.findOneAndDelete({ _id: new ObjectID(recordId) }, (err, result) => {
    if (err) {
      console.error('Error al eliminar el registro:', err);
      res.status(500).send('Error de servidor');
      return;
```

```
        }
        res.json(result.value);
    });
});

app.listen(port, () => {
    console.log(`Servidor escuchando en el puerto ${port}`);
});
```