https://www.tutorialkart.com/nodejs/nodejs-mysql-update/

// include http module in the file

var http = require('http');

// create a server

http.createServer(function (req, res) {

    // http header

    // 200 - is the OK message

    // to respond with html content, 'Content-Type' should be 'text/html'

    res.writeHead(200, {'Content-Type': 'text/html'});

    res.write('Node.js says hello!'); //write a response to the client

    res.end(); //end the response

}).listen(9000); //the server object listens on port 9000

1. Include the module

The first step to extend a module is to include the module itself using require function.

|  |
| --- |
| var newMod = require('<module\_name>'); |

We have retrieved the module to a variable.

2. Add function to the module variable

Using variable to the module,newMod , add a new function to it using following syntax.

|  |
| --- |
| newMod.<newFunctionName> = function(function\_parameters) {      // function body    }; |

You may add as many number of new functions to the module as per your requirement. Any modifications to the module variable does not affect the actual module in its original form.

3. Re-export the module

You have to re-export the module for the new added functionalities to take effect.

|  |
| --- |
| module.exports = newMod; |

Now, you may use the variable to the module,newMod , for calling new functionalities added.

Extend or add functions to Node.js module

// include the module that you like extend

var fs = require('fs');

// add a new function, printMessage(), to the module

fs.printMessage = function(str){

    console.log("Message from newly added function to the module");

    console.log(str);

}

// re-export the module for changes to take effect

module.exports = fs

// you may use the newly added function

fs.printMessage("Success");

To write a JSON Object to a local file, following is a step-by-step guide :

1. Stringify JSON Object. UseJSON.stringify(jsonObject)  to convert JSON Object to JSON String.
2. Write the stringified object to file using fs.writeFile() function of [Node FS](https://www.tutorialkart.com/nodejs/node-fs/) module.

// file system module to perform file operations

const fs = require('fs');

// json data

var jsonData = '{"persons":[{"name":"John","city":"New York"},{"name":"Phil","city":"Ohio"}]}';

// parse json

var jsonObj = JSON.parse(jsonData);

console.log(jsonObj);

// stringify JSON Object

var jsonContent = JSON.stringify(jsonObj);

console.log(jsonContent);

fs.writeFile("output.json", jsonContent, 'utf8', function (err) {

    if (err) {

        console.log("An error occured while writing JSON Object to File.");

        return console.log(err);

    }

    console.log("JSON file has been saved.");

});

Example 1 – Node.js JSON Parsing

In this example, we will use JSON.parse() function to parse the string jsonData. Also, we will access the elements from JSON Object using DOT operator.

**nodejs-parse-json.js**

|  |
| --- |
| // json data  var jsonData = '{"persons":[{"name":"John","city":"New York"},{"name":"Phil","city":"Ohio"}]}';    // parse json  var jsonParsed = JSON.parse(jsonData);    // access elements  console.log(jsonParsed.persons[0].name); |

Example 2 – Node.js Parse JSON File

In this example, we shall [read a File](https://www.tutorialkart.com/nodejs/read-a-file-in-nodejs-using-fs-module/) containing JSON data to a variable and parse that data.

Consider following JSON File. We shall read this file as string to a variable.

**sample.json**

|  |
| --- |
| {      "persons": [{              "name": "John",              "city": "Kochi",              "phone": {                  "office": "040-528-1258",                  "home": "9952685471"              }            },          {              "name": "Phil",              "city": "Varkazha",              "phone": {                  "office": "040-528-8569",                  "home": "7955555472"              }          }      ]  } |

**nodejs-parse-json-file.js**

|  |
| --- |
| // include file system module  var fs = require('fs');    // read file sample.json file  fs.readFile('sample.json',      // callback function that is called when reading file is done      function(err, data) {          // json data          var jsonData = data;            // parse json          var jsonParsed = JSON.parse(jsonData);            // access elements          console.log(jsonParsed.persons[0].name + "'s office phone number is "  + jsonParsed.persons[0].phone.office);          console.log(jsonParsed.persons[1].name + " is from " +  jsonParsed.persons[0].city);  }); |

Example 1 – Write data to file in Node.js

In this example, we shall write content, “Hello !” , to a text file sample.txt.

**nodejs-write-to-file-example.js**

|  |
| --- |
| // include file system module    var fs = require('fs');    var data = "Hello !"    // write data to file sample.html  fs.writeFile('sample.txt', data,      // callback function that is called after writing file is done      function(err) {          if (err) throw err;          // if no error          console.log("Data is written to file successfully.")  }); |

Example 1 – Node.js Append data to file asynchronously using appendFile()

To append data to a file asynchronously in Node.js, use appendFile() function of Node FS as shown below.

**nodejs-append-to-file-example.js**

|  |
| --- |
| // Example Node.js program to append data to file  var fs = require('fs');    var data = "\nLearn Node.js with the help of well built Node.js Tutorial.";    // append data to file  fs.appendFile('sample.txt',data, 'utf8',      // callback function      function(err) {          if (err) throw err;          // if no error          console.log("Data is appended to file successfully.")  }); |

Example 2 – Node.js Append data to file synchronously using appendFileSyc()

To append data to a file synchronously in Node.js, use appendFileSync() function of Node FS as shown below.

**nodejs-append-to-file-example-2.js**

|  |
| --- |
| // Example Node.js program to append data to file  var fs = require('fs');    var data = "\nLearn Node.js with the help of well built Node.js Tutorial.";    // append data to file  fs.appendFileSync('sample.txt',data, 'utf8');  console.log("Data is appended to file successfully.") |

Example to Rename file Synchronously

In this example, we will rename a file synchronously, from sample\_old.txt to sample.txt.

To rename a file synchronously in Node.js using Node FS, use renameSync() function as shown below.

**nodejs-rename-file.js**

|  |
| --- |
| var fs = require('fs');    fs.renameSync('sample.txt', 'sample\_old.txt');  console.log('File Renamed.'); |

Q1 - What is Node.js?

Node.js is :

* Open Source (Source code of Node.js is available openly and can be modified and built)
* Cross Platform (Works on different operating systems)
* Asynchronous
* Event Driven (Web Requests are considered as events)
* JavaScript Runtime Environment (JavaScript is run outside the browser)

Q2 - Which is the scripting/programming language used for Node.js application programming?

The answer is one word. JavaScript.

Q3 - How is Node.js different from previous server side programming frameworks?

Previous frameworks were not event driven. Node.js is a single thread driven.

Q4 - What are the features of Node.js?

Some of the important features of Node.js are :

* Asynchronous and Event Driven
* Code Execution is very fast since Node.js is built on Chrome’s V8 JavaScript Engine.
* Highly Scalable because of Event Looping.

### Q5 - How is event-driven programming?

Requests to the Node.js are considered as events. When a request is made, Node.js server accepts the request,  attaches it to a callback,  and starts preparing the response. When the response is being prepared, the server does not wait; it takes in other requests from clients. Whenever a response is ready, it is served via callback function. Thus everything is considered an event,

Q6 - What is MEAN stack? What is the place of Node.js in the MEAN stack?

MEAN is a JavaScript software stack. It is used for building dynamic websites and web applications. MEAN Stack components are :

1. **M**ongoDB
2. **E**xpress.js
3. **A**ngular.js or Angular
4. **N**ode.js

### Q7 - How do you install a module in node?

A module is installed in node using NPM command line interface, npm. The command to install a module is : **npm install** <**package**>

### Q8 - What is npm?

npm stands for Node Package Manager. npm is the package manager for JavaScript.

What is the command for importing a package?

require command is used to include a package/module in a script file.

Example to include fs module in the script file is :

|  |
| --- |
| var fs = require('fs'); |

How to add new functionalities to a module?

New functions could be added to the existing modules. The step by step process is :

1. Include the module in the script file using require statement.
2. Add a function to the module using dot (.) operator.
3. Export the module for the changes to take effect.
4. How to exit from a Node.js Script?
5. The execution of a Node.js script happens sequentially. For some reason, if you want to exit, the built-in process module has exit() method.

|  |
| --- |
| process.exit(-1); |

How do you redirect a url to other in Node.js?

http module can be used to setup an url redirect.

In the following example, we created an HTTP server. When the server gets a request for a resource, we can send a redirect response back to the browser using response.writeHead() function. The response code should be 301 for a redirect and the redirect url is provided as well in writeHead() function.

|  |
| --- |
| var http = require('http');  var fs = require('fs');    // create a http server  http.createServer(function (req, res) {        if (req.url == '/page-c.html') {          // redirect to page-b.html with 301 (Moved Permanently) HTTP code in the response          res.writeHead(301, { "Location": "[http://](NULL)" + req.headers['host'] + '/page-b.html' });          return res.end();      } else {          // for other URLs, try responding with the page          console.log(req.url)          // read requested file          fs.readFile(req.url.substring(1),              function(err, data) {                  if (err) throw err;                  res.writeHead(200);                  res.write(data.toString('utf8'));                  return res.end();          });      }  }).listen(8085); |

How do you connect your Node.js application to MySQL?

From Node.js, we can connect to MySQL server. We can use mysql module.

1. Include mysql module in the script file.
2. Create a connection variable with the information : MySQL Server IP, Port, Username, Password.
3. Call connect method using connection variable. If the setup is fine, the callback function should receive a null error object.

An example Node.js script to connect to MySQL server :

|  |
| --- |
| // include mysql module  var mysql = require('mysql');    // create a connection variable with the details required  var con = mysql.createConnection({    host: "localhost",    // ip address of server running mysql    user: "arjun",    // user name to your mysql database    password: "password"    // corresponding password  });    // connect to the database.  con.connect(function(err) {    if (err) throw err;    console.log("Connected!");  }); |

Example 1 – MySQL SELECT FROM Query via Node.js

In this example, we select all the rows of the MySQL table. The result contains all the rows returned by the SELECT Query.

**example.js**

|  |
| --- |
| // Node.js MySQL SELECT FROM query Example  // include mysql module  var mysql = require('mysql');    // create a connection variable with the required details  var con = mysql.createConnection({    host: "localhost",    // ip address of server running mysql    user: "arjun",    // user name to your mysql database    password: "password", // corresponding password    database: "studentsDB" // use the specified database  });    // make to connection to the database.  con.connect(function(err) {    if (err) throw err;    // if connection is successful    con.query("SELECT \* FROM students", function (err, result, fields) {      // if any error while executing above query, throw error      if (err) throw err;      // if there is no error, you have the result      console.log(result);    });  }); |

**Output**

Access **Result** Object of MySQL SELECT FROM Query via Node.js

In this example, we will access rows from Result Object using index, columns and DOT operator.

**example.js**

|  |
| --- |
| // Node.js MySQL SELECT FROM query Example  // include mysql module  var mysql = require('mysql');    // create a connection variable with the required details  var con = mysql.createConnection({    host: "localhost",    // ip address of server running mysql    user: "arjun",    // user name to your mysql database    password: "password", // corresponding password    database: "studentsDB" // use the specified database  });    // make to connection to the database.  con.connect(function(err) {    if (err) throw err;    // if connection is successful    con.query("SELECT \* FROM students", function (err, result, fields) {      // if any error while executing above query, throw error      if (err) throw err;      // if there is no error, you have the result      // iterate for all the rows in result      Object.keys(result).forEach(function(key) {        var row = result[key];        console.log(row.name)      });    });  }); |

**Output**

 INSERT Row into MySQL Table via Node.js

In this example, we will connect to MySQL database, and insert a record into students table.

**insertIntoExample.js**

|  |
| --- |
| // include mysql module  var mysql = require('mysql');    // create a connection variable with the required details  var con = mysql.createConnection({    host: "localhost",    // ip address of server running mysql    user: "arjun",    // user name to your mysql database    password: "password", // corresponding password    database: "studentsDB" // use the specified database  });    // make to connection to the database.  con.connect(function(err) {    if (err) throw err;    // if connection is successful    con.query("INSERT INTO students (name,rollno,marks) values ('Anisha',12,95)", function (err, result, fields) {      // if any error while executing above query, throw error      if (err) throw err;      // if there is no error, you have the result      console.log(result);    });  }); |

INSERT Multiple Rows into MySQL Table via Node.js

In this example, we will connect to MySQL database, and insert three records into students table in a single statement.

**insertMulIntoExample.js**

|  |
| --- |
| // include mysql module  var mysql = require('mysql');    // create a connection variable with the required details  var con = mysql.createConnection({    host: "localhost",    // ip address of server running mysql    user: "arjun",    // user name to your mysql database    password: "password", // corresponding password    database: "studentsDB" // use the specified database  });    // make to connection to the database.  con.connect(function(err) {    if (err) throw err;    // if connection is successful    var records = [      ['Miley', 13, 85],      ['Jobin', 14, 87],      ['Amy', 15, 74]    ];    con.query("INSERT INTO students (name,rollno,marks) VALUES ?", [records], function (err, result, fields) {      // if any error while executing above query, throw error      if (err) throw err;      // if there is no error, you have the result      console.log(result);    });  }); |

Update Row(s) of MySQL Table via Node.js

Consider the case that due to manual error, records in the **students** table are inserted with marks as 74 instead of 84. Now we shall execute a MySQL Update Query that updates **marks** column with value 84 where there is a value of 74.

**UpdateRecordsFiltered.js**

|  |
| --- |
| // include mysql module  var mysql = require('mysql');    // create a connection variable with the required details  var con = mysql.createConnection({    host: "localhost",    // ip address of server running mysql    user: "arjun",    // user name to your mysql database    password: "password", // corresponding password    database: "studentsDB" // use the specified database  });    // make to connection to the database.  con.connect(function(err) {    if (err) throw err;    // if connection is successful    con.query("UPDATE students SET marks=84 WHERE marks=74", function (err, result, fields) {      // if any error while executing above query, throw error      if (err) throw err;      // if there is no error, you have the result      console.log(result);    });  }); |

// include mysql module

var mysql = require('mysql');

// create a connection variable with the required details

var con = mysql.createConnection({

  host: "localhost",    // ip address of server running mysql

  user: "arjun",    // user name to your mysql database

  password: "password", // corresponding password

  database: "studentsDB" // use the specified database

});

// make connection to the database.

con.connect(function(err) {

  if (err) throw err;

  // if connection is successful

  con.query("DELETE FROM students WHERE rollno>10", function (err, result, fields) {

    // if any error while executing above query, throw error

    if (err) throw err;

    // if there is no error, you have the result

    console.log(result);

  });

});