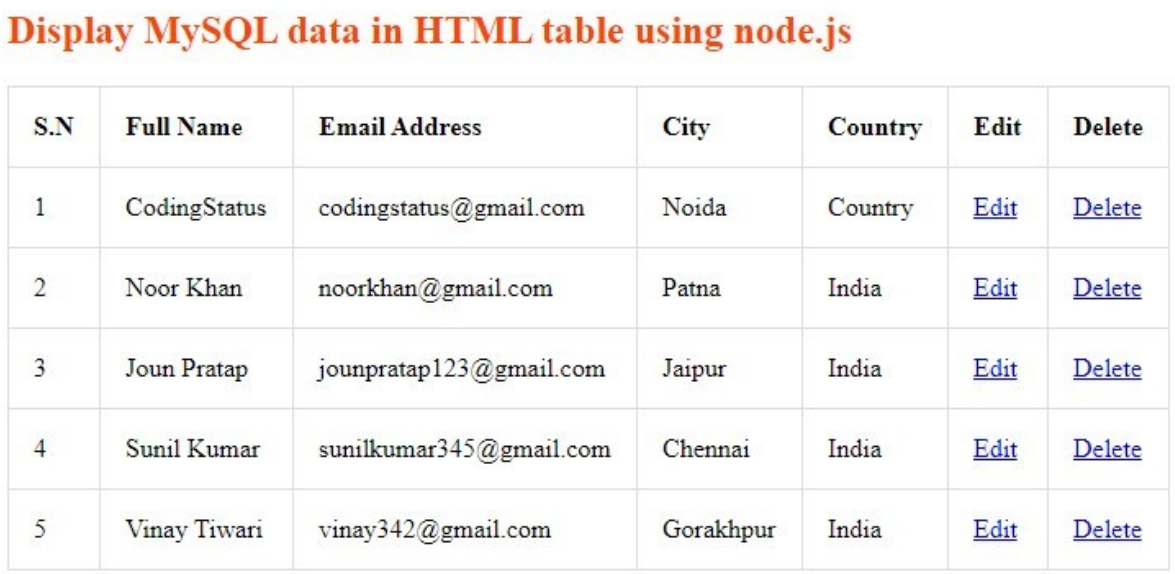
How to display Data from MySQL database table in Node.js

December 21, 2021 By Md Nurullah

In this tutorial, I will show you how to display data from the MySQL database table using Node.js. Even you will learn this tutorial with a simple example. In this example, All the data will display in the HTML table in the proper format. So, read all the following steps and use them in your project.



Display MySQL Data in HTML Table Using Node.js

Before getting started, You must insert data into the MySQL database using node.js. Even make sure that Database Name is 'nodeapp' and table name is 'users'. If you want to use your own database & table name then you can.

1. Install Express Application

You have to Install Basic Express Application like the following project folder structure


```
myapp/
  |__bin
  |__node_modules
  |__public
  |__routes/
  |    |__index.js
  |    |__users.js
  |__views/
  |    |__index.ejs
  |    |__users.ejs
  |    |__user-list.ejs
  |__app.js
  |__database.js
  |__package-lock.json
  |__package.json
```

2. Connect Node.js App to MySQL Database

First of all, you have to connect the node.js app to the MySQL database with the connection credentials like lost, username, password & database name

File Name – database.js

```
var mysql = require('mysql');
var conn = mysql.createConnection({
  host: 'localhost', // Replace with your host name
  user: 'root',      // Replace with your database username
  password: '',      // Replace with your database password
  database: 'nodeapp' // // Replace with your database Name
});
conn.connect(function(err) {
```

```
  if (err) throw err;
  console.log('Database is connected successfully !');
});
module.exports = conn;
```

3. Create a Route to Fetch Data

You have to configure the following steps to create a route for fetching data using MySQL in Node.js –

- Include the database connection file `database.js`
- Create a route `/user-list` to fetch data from the `users` table
- Write a SQL query to fetch data from the database.
- Pass the fetched data into the view file in the form of `userData`
- At last. export router.

File Name – users.js

```
var express = require('express');
var router = express.Router();
var db=require('../database');
// another routes also appear here
// this script to fetch data from MySQL databse table
router.get('/user-list', function(req, res, next) {
    var sql='SELECT * FROM users';
    db.query(sql, function (err, data, fields) {
    if (err) throw err;
    res.render('user-list', { title: 'User List', userData: data});
  });
});
module.exports = router;
```

If you want to create your own route URL according to your project requirement then you can. But make sure that all the scripts must be correct. otherwise, your app will not work properly.

4. Load Route into the root file

You have to include the created route in the app.js. If you forget to include this file, your app will not work. You will get an error when you

try to execute your script.

File Name – app.js

```
var createError = require('http-errors');
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');
var indexRouter = require('./routes/index');
var usersRouter = require('./routes/users');
;
var app = express();

// view engine setup
app.use('/', indexRouter);
app.use('/users', usersRouter);



app.set('views', path.join(__dirname, 'views'));

app.set('view engine', 'ejs');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};
```

```
  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;
```

5. Create HTML Table to Display data

Configure the following steps to create an HTML table for displaying data –

- Create a file `user-list.ejs` in the `views` folder.
- Display data with the help of passing variable `userData` from the route.

File Name – user-list.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Fetch using MySQL and Node.js</title>
</head>
<body>
    <div class="table-data">
<h2>Display Data using Node.js & MySQL</h2>
    <table border="1">
        <tr>
            <th>S.N</th>
            <th>Full Name</th>
            <th>Email Address</th>
            <th>City</th>
            <th>Country</th>
            <th>Edit</th>
            <th>Delete</th>
        </tr>

        <%
        if(userData.length!=0){
        var i=1;
        userData.forEach(function(data){
        %>
        <tr>
```

```
                <td><%=i; %></td>
                <td><%=data.fullName %></td>
                <td><%=data.emailAddress %></td>
                <td><%=data.city %></td>
                <td><%=data.country %></td>
                <td><a href="/users/edit/<%=data.id%>">Edit</a></td>
                <td><a href="/users/delete/<%=data.id%>">Delete</a></td>
            </tr>
            <%  i++; }) %>
            <% } else{ %>
                <tr>
                    <td colspan="7">No Data Found</td>
                </tr>
            <% } %>
        </table>
        </div>
    </body>
    </html>
```

6. Run Node.js Code to display data in the HTML Table

To Run Node.js code for displaying data in the HTML table, you will have to do the following things –

- Start your Node.js server using `npm start` the command
- Enter the following URL into your web browser

```
http://localhost:3000/users/user-list
```

In this tutorial, I have taught you to display data in an HTML table with simple records of a table. If are a beginner then you should execute this script as it is given. After that, try to implement it with another kind of largest data/records. If you are experienced and you need only its query then you can use it directly in your project.

My Suggestion

Dear developers, I hope you have understood the above script, Now you are able to display data from the MySQL database table in Node.js. Even you can display another table of data by using the above steps.

If you have any questions or suggestions regarding Node.js. You can directly ask me through the below comment box.

Filed Under: Node.js

Hey there, Welcome to CodingStatus. My Name is Md Nurullah from Bihar, India. I'm a Software Engineer. I have been working in the Web Technology field for 3 years. Here, I blog about Web Development & Designing. Even I help developers to build the best Web Applications.

Search the site ...

LATEST TUTORIALS

- JavaScript Color Picker Chrome Extension
- To Do List Using PHP and MySQL
- How to Create JavaScript Accordion
- Upload Multiple Files to Store in MySQL Database Using PHP
- Ajax File Upload – Without Refreshing Page

POPULAR TUTORIALS

- How to Download and Install Node.js on Windows with NPM
- How to Display Image From MySQL Database in Node.js
- How to display Data from MySQL database table in Node.js
- Upload Multiple Files Using Multer in Node.js and Express
- Node.js File Upload Using Express

Multer

CATEGORIES

- Ajax (10)
- Django (5)
- HTML (4)
- Installation (3)
- Interview Questions (5)
- JavaScript (19)
- jQuery (11)
- Laravel (2)
- Node.js (23)
- PHP (39)
- ReactJS (35)
- SQL (12)
- Tips (7)