

Node.js MySQL SELECT FROM Query Examples

Learn Node.js MySQL SELECT FROM query to access data of a MySQL Table.

Node.js MySQL SELECT FROM query

MySQL SELECT Query is used to select some of the records (with some of their properties if required) of a table.

In this Node.js Tutorial, we shall learn to access data of a table using following Node.js examples

- [Example to MySQL SELECT FROM query](#)
- [Example to select only some of the columns](#)
- [Example to use **Result** Object of MySQL SELECT FROM query](#)
- [Example to use **Fields** Object of MySQL SELECT FROM query](#)

We shall use the following MySQL Table, in the examples of this section [DATABASE : studentsDB, Table: students]

studentsDB.students table

```
mysql> select * from
```

```
mysql> select * from students;
+-----+-----+-----+
| name | rollno | marks |
+-----+-----+-----+
| John | 1 | 74 |
| Arjun | 2 | 74 |
| Prasanth | 3 | 77 |
| Adarsh | 4 | 78 |
| Raja | 5 | 94 |
| Sai | 6 | 84 |
| Ross | 7 | 54 |
| Monica | 8 | 86 |
| Lee | 9 | 98 |
| Bruce | 10 | 92 |
| Sukumar | 11 | 99 |
+-----+-----+-----+
11 rows in set (0.01 sec)
```

Example to MySQL SELECT FROM query

selectFromTable.js Simple example to MySQL SELECT FROM query

```
// Node.js MySQL
SELECT FROM query
```

```
// 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);
  });
});
```

```
$ node
selectFromTable.js
```

```
$ node selectFromTable.js
[ RowDataPacket { name: 'John', rollno: 1, marks: 74 },
  RowDataPacket { name: 'Arjun', rollno: 2, marks: 74 },
  RowDataPacket { name: 'Prasanth', rollno: 3, marks: 77 },
  RowDataPacket { name: 'Adarsh', rollno: 4, marks: 78 },
  RowDataPacket { name: 'Raja', rollno: 5, marks: 94 },
  RowDataPacket { name: 'Sai', rollno: 6, marks: 84 },
  RowDataPacket { name: 'Ross', rollno: 7, marks: 54 },
  RowDataPacket { name: 'Monica', rollno: 8, marks: 86 },
  RowDataPacket { name: 'Lee', rollno: 9, marks: 98 },
  RowDataPacket { name: 'Bruce', rollno: 10, marks: 92 },
  RowDataPacket { name: 'Sukumar', rollno: 11, marks: 99 } ]
```

Example to select only some of the columns

We shall select only two columns, name and marks

```
// Node.js MySQL  
SELECT FROM query
```

```
// 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 name,marks 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);  
  });  
});
```

```
$ node  
selectColumns.js
```

```
$ node selectColumns.js  
[ RowDataPacket { name: 'John', marks: 74 },  
  RowDataPacket { name: 'Arjun', marks: 74 },  
  RowDataPacket { name: 'Prasanth', marks: 77 },  
  RowDataPacket { name: 'Adarsh', marks: 78 },  
  RowDataPacket { name: 'Raja', marks: 94 },  
  RowDataPacket { name: 'Sai', marks: 84 },  
  RowDataPacket { name: 'Ross', marks: 54 },  
  RowDataPacket { name: 'Monica', marks: 86 },  
  RowDataPacket { name: 'Lee', marks: 98 },  
  RowDataPacket { name: 'Bruce', marks: 92 },  
  RowDataPacket { name: 'Sukumar', marks: 99 } ]
```

Example to use **Result** Object of MySQL SELECT FROM query

You may access rows using index and columns using dot operator.

selectUseResultObject.js - Access rows and column data of result object

```
// Node.js MySQL
SELECT FROM query

// 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)
    });
  });
});
```

```
$ node
selectUseResultObject.js
```

```
$ node selectUseResultObject.js
```

John

Arjun

Prasanth

Adarsh

Raja

Sai

Ross

Monica

Lee

Bruce

Sukumar

Example to use **Fields** Object of MySQL SELECT FROM query

Fields contain information about columns of table. Each field contains all information about a column.

selectUseFieldsObject.js - Example for the usage of fields

```
// Node.js MySQL  
SELECT FROM query
```

```
// 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 fields object
    // iterate for all the rows in fields object
    Object.keys(fields).forEach(function(key) {
      var field = fields[key];
      console.log(field)
    });
  });
});
```

```
$ node
selectUseFieldsObject.js
```

```
$ node selectUseFieldsObject.js
FieldPacket {
  catalog: 'def',
  db: 'studentsDB',
  table: 'students',
  orgTable: 'students',
  name: 'name',
  orgName: 'name',
  charsetNr: 33,
  length: 150,
  type: 253,
  flags: 0,
  decimals: 0,
  default: undefined,
  zeroFill: false.
```

```
protocol41: true }
FieldPacket {
  catalog: 'def',
  db: 'studentsDB',
  table: 'students',
  orgTable: 'students',
  name: 'rollno',
  orgName: 'rollno',
  charsetNr: 63,
  length: 11,
  type: 3,
  flags: 0,
  decimals: 0,
  default: undefined,
  zeroFill: false,
  protocol41: true }
FieldPacket {
  catalog: 'def',
  db: 'studentsDB',
  table: 'students',
  orgTable: 'students',
  name: 'marks',
  orgName: 'marks',
  charsetNr: 63,
  length: 11,
  type: 3,
  flags: 0,
  decimals: 0,
  default: undefined,
  zeroFill: false,
  protocol41: true }
```

You may use the elements of a field object using dot operator. Example field.catalog, field.name, field.type, etc.

Conclusion :

In this [Node.js Tutorial](#) – [Node.js MySQL](#) – Node.js MySQL SELECT FROM query, we have learnt to fetch records of table from MySQL database, and to use result object and fields object.

Node.js

» [Node.js Tutorial](#)

Get Started With Node.js

» [Install Node.js Ubuntu Linux](#)

» [Install Node.js Windows](#)

- Node.js - Basic Example
- Node.js - Command Line Arguments
- Node.js - Modules
- Node.js - Create a module
- Node.js - Add new functions to Module
- Node.js - Override functions of Module
- Node.js - Callback Function
- Node.js - forEach

Express.js

- Express.js Tutorial
- What is Express.js?
- Express.js Application Example
- Install Express.js
- Express.js Routes
- Express.js Middleware
- Express.js Router

Node.js Buffers

- Node.js Buffer - Create, Write, Read
- Node.js Buffer - Length
- Node.js - Convert JSON to Buffer
- Node.js - Array to Buffer

Node.js HTTP

- Node.js - Create HTTP Web Server
- Node.js - Redirect URL

Node.js MySQL

- Node.js MySQL
- Node.js MySQL - Connect to MySQL Database
- Node.js MySQL - SELECT FROM
- Node.js MySQL - SELECT WHERE
- Node.js MySQL - ORDER BY
- Node.js MySQL - INSERT INTO
- Node.js MySQL - UPDATE
- Node.js MySQL - DELETE
- Node.js MySQL - Result Object

» [Node.js MySQL - Result Object](#)

Node.js MongoDB

» [Node.js MongoDB](#)

» [Node.js - Connect to MongoDB](#)

» [Node.js - Create Database in MongoDB](#)

» [Node.js - Drop Database in MongoDB](#)

» [Node.js - Create Collection in MongoDB](#)

» [Node.js - Delete Collection in MongoDB](#)

» [Node.js - Insert Documents to MongoDB Collection](#)

» [MongoError: failed to connect to server](#)

Node.js Mongoose

» [Node.js Mongoose Tutorial](#)

» [Node.js Mongoose - Installation](#)

» [Node.js Mongoose - Connect to MongoDB](#)

» [Node.js Mongoose - Define a Model](#)

» [Node.js Mongoose - Insert Single Document to MongoDB](#)

» [Node.js Mongoose - Insert Multiple Documents to MongoDB](#)

Node.js URL

» [Node.js - Parse URL parameters](#)

Node.js FS (File System)

» [Node FS](#)

» [Node FS - Read a File](#)

» [Node FS - Create a File](#)

» [Node FS - Write to a File](#)

» [Node FS - Append to a File](#)

» [Node FS - Rename a File](#)

» [Node FS - Delete a File](#)

» [Node FS Extra - Copy a Folder](#)

Node.js JSON

» [Node.js Parse JSON](#)

» [Node.js Write JSON Object to File](#)

Node.js Error Handling

» [Node.js Try Catch](#)

Node.js Events

Node.js Examples

- » [Node.js Examples](#)
- » [Node.js - Handle Get Requests](#)
- » [Node.js Example - Upload files to Node.js server](#)

Useful Resources

- » [Node.js Interview Questions](#)
- » [How to Learn Programming](#)