**UNIT-V**

**MongoDB:** Access MongoDB in Node.js, Connecting and Creating Database in MongoDB, Insert Documents, Update Documents, Delete Documents, Query Database.

**Access MongoDB in Node.js**

Learn how to access document-based database MongoDB using Node.js in this section.

In order to access MongoDB database, we need to install MongoDB drivers. To install native [mongodb](https://www.npmjs.com/package/mongodb" \t "_blank) drivers using NPM, open command prompt and write the following command to install MongoDB driver in your application.

npm install mongodb --save

This will include mongodb folder inside node\_modules folder. Now, start the MongoDB server using the following command. (Assuming that your MongoDB database is at C:\MyNodeJSConsoleApp\MyMongoDB folder.)

mongod -dbpath C:\MyNodeJSConsoleApp\MyMongoDB

Connecting MongoDB

The following example demonstrates connecting to the local MongoDB database.

app.js

 Copy

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

// Connect to the db

MongoClient.connect("mongodb://localhost:27017/MyDb", function (err, db) {

if(err) throw err;

//Write databse Insert/Update/Query code here..

});

In the above example, we have imported mongodb module (native drivers) and got the reference of MongoClient object. Then we used MongoClient.connect() method to get the reference of specified MongoDB database. The specified URL "mongodb://localhost:27017/MyDb" points to your local MongoDB database created in MyMongoDB folder. The connect() method returns the database reference if the specified database is already exists, otherwise it creates a new database.

Now you can write insert/update or query the MongoDB database in the callback function of the connect() method using db parameter.

# Node.js MongoDB Create Database

To create a database in MongoDB, First create a MongoClient object and specify a connection URL with the correct ip address and the name of the database which you want to create.

**Example**

Create a folder named "MongoDatabase" as a database. Suppose you create it on Desktop. Create a js file named "createdatabase.js" within that folder and having the following code:

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

var url = "mongodb://localhost:27017/MongoDatabase";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

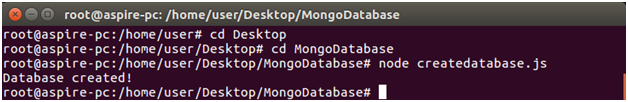
console.log("Database created!");

db.close();

});

Now open the command terminal and set the path where MongoDatabase exists. Now execute the following command:

Node createdatabase.js



Now database is created.

**Insert Documents**

The following example demonstrates inserting documents into MongoDB database.

app.js

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

// Connect to the db

MongoClient.connect("mongodb://localhost:27017/MyDb", function (err, db) {

db.collection('Persons', function (err, collection) {

collection.insert({ id: 1, firstName: 'Steve', lastName: 'Jobs' });

collection.insert({ id: 2, firstName: 'Bill', lastName: 'Gates' });

collection.insert({ id: 3, firstName: 'James', lastName: 'Bond' });

db.collection('Persons').count(function (err, count) {

if (err) throw err;

console.log('Total Rows: ' + count);

});

});

});

In the above example, db.collection() method creates or gets the reference of the specified collection. Collection is similar to table in relational database. We created a collection called Persons in the above example and insert three documents (rows) in it. After that, we display the count of total documents stored in the collection.

Running the above example displays the following result.

**Update/Delete Documents**

The following example demonstrates updating or deleting an existing documents(records).

app.js

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

// Connect to the db

MongoClient.connect("mongodb://localhost:27017/MyDb", function (err, db) {

db.collection('Persons', function (err, collection) {

collection.update({id: 1}, { $set: { firstName: 'James', lastName: 'Gosling'} }, {w:1},

function(err, result){

if(err) throw err;

console.log('Document Updated Successfully');

});

collection.remove({id:2}, {w:1}, function(err, result) {

if(err) throw err;

console.log('Document Removed Successfully');

});

});

});

**Query Database**

The following example demonstrates executing a query in the MongoDB database.

app.js

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

// Connect to the db

MongoClient.connect("mongodb://localhost:27017/MyDb", function (err, db) {

db.collection('Persons', function (err, collection) {

collection.find().toArray(function(err, items) {

if(err) throw err;

console.log(items);

});

});

});

So, in this way you can connect and access MongoDB database.