# How to connect to SQL Server database in Node.js using Tedious

## Introduction

Tedious is a Node package that provides an implementation of the [TDS protocol](http://msdn.microsoft.com/en-us/library/dd304523.aspx), which is used to interact with instances of Microsoft’s SQL Server.

This sample demonstrate how to connect to SQL Server database in Node.js using Tedious.

## Sample prerequisites

To open and run this sample, ensure that the following requisites have been met:

* Node.js 6.6.0 or above.
* NPM is installed (In default case, it have been installed before you installed Node.js).
* You should have a SQL database, and have below structure.

CREATE TABLE Student(

ID int identity primary key,

Name nvarchar(50),

Age int

)

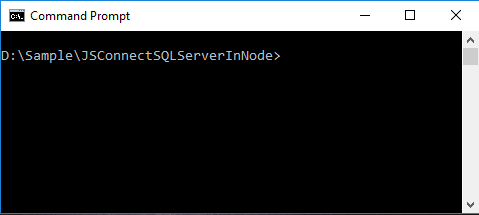
INSERT INTO Student VALUES('Bear',18)

INSERT INTO Student VALUES('Frank',20)

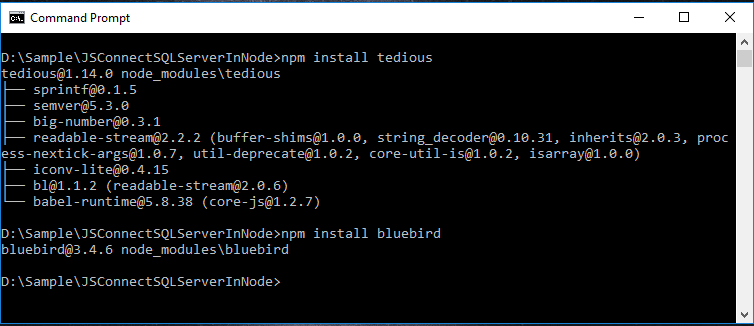
## Building the sample

**Restore the library**

* Open the Command Prompt window and navigate to sample location folder, for example, the sample location is D:\Sample\JSConnectSQLServerInNode.

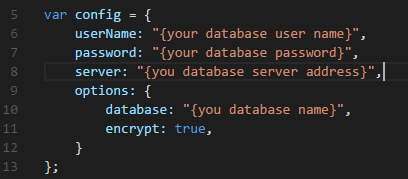


* Type under command to restore library.
  + npm install tedious
  + npm install bluebird



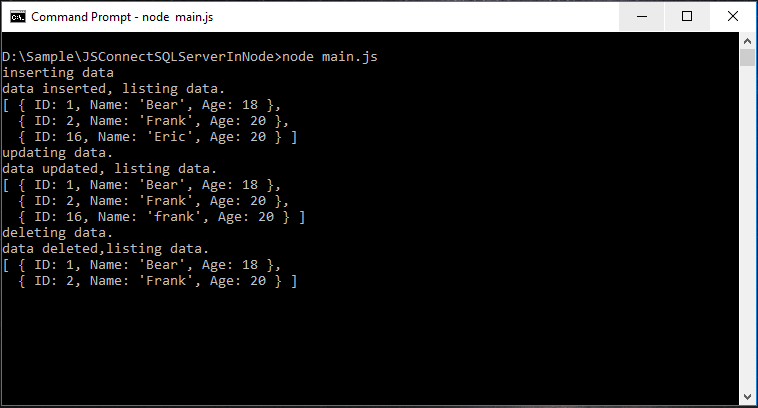
**Config Connection portal**

* Open the file main.js
* Local to config section
* Fill your SQL connection portal



## Running the sample

* Open the Command Prompt window and navigate to sample location folder, for example, the sample location is D:\Sample\JSConnectSQLServerInNode.
* Type command: node main.js, the program will connect to the SQL Server, and finish some CURD actions.



## Using the code

The base use:

var Connection = require('tedious').Connection;

var Request = require('tedious').Request;

var config = {

userName: "{your database user name}",

password: "{your database password}",

server: "{you database server address}",

options: {

database: "{you database name}",

encrypt: true,

}

};

var connection = new Tedious.Connection(config);

connection.on("connect",function(err){

var result = [];

var request = new Request("select \* form student",function(err,count,rows){

console.log(result);

});

request.on("row", function (columns) {

var item = {};

columns.forEach(function (column) {

item[column.metadata.colName] = column.value;

});

result.push(item);

});

})

**But it’s really unfriendly for coding, so let’s improve this.**

**Build a SQL helper file name msSqlConnecter.js, with below code:**

var Tedious = require("tedious");

var Promise = require("bluebird");

module.exports = {

msSqlConnecter: function (config) {

var currentConnect = this;

currentConnect.config = config;

currentConnect.errorHandler;

currentConnect.connectedHandler;

currentConnect.connection;

currentConnect.onConnected = function (callback) {

currentConnect.connectedHandler = callback;

return currentConnect;

};

currentConnect.onError = function (callback) {

currentConnect.errorHandler = callback;

return currentConnect;

};

currentConnect.Request = function (sql) {

var currentRequest = this;

currentRequest.sql = sql;

currentRequest.params = [];

currentRequest.result = [];

currentRequest.errorHandler;

currentRequest.onComplateHandler;

currentRequest.addParam = function (key, type, value) {

currentRequest.params.push({ key: key, type: type, value: value });

return currentRequest;

}

currentRequest.Run = function () {

var request = new Tedious.Request(currentRequest.sql, function (err, rowCount, rows) {

if (err) {

currentRequest.errorHandler(err);

}

else {

currentRequest.onComplateHandler(rowCount, currentRequest.result);

}

});

request.on("row", function (columns) {

var item = {};

columns.forEach(function (column) {

item[column.metadata.colName] = column.value;

});

currentRequest.result.push(item);

});

for (var i in currentRequest.params) {

var item = currentRequest.params[i];

request.addParameter(item.key, item.type, item.value);

}

currentConnect.connection.execSql(request);

return currentRequest;

};

currentRequest.onError = function (callback) {

currentRequest.errorHandler = callback;

return currentRequest;

};

currentRequest.onComplate = function (callback) {

currentRequest.onComplateHandler = callback;

return currentRequest;

};

}

currentConnect.connect = function () {

var connection = new Tedious.Connection(config);

currentConnect.connection = connection;

return Promise.promisify(connection.on.bind(connection))("connect");

}

}

}

**And How to use this SQL Helper.**

**In Main.js**

var TYPES = require("tedious").TYPES;

var msSqlConnecter = require("./msSqlConnecter");

var config = {

userName: "{your database user name}",

password: "{your database password}",

server: "{you database server address}",

options: {

database: "{you database name}",

encrypt: true,

}

};

function insert(callback) {

//when insert

var con = new msSqlConnecter.msSqlConnecter(config);

con.connect().then(function () {

new con.Request("insert into student values(@name,@age)")

.addParam("name", TYPES.VarChar, "Eric")

.addParam("age", TYPES.Int, 20)

.onComplate(function (count) {

if (callback)

callback(count);

})

.onError(function (err) {

console.log(err);

})

.Run();

}).catch(function (ex) {

console.log(ex);

});

}

function queryAll(callback) {

var con = new msSqlConnecter.msSqlConnecter(config);

con.connect().then(function () {

new con.Request("select \* from student")

.onComplate(function (count, datas) {

if (callback)

callback(datas);

})

.onError(function (err) {

console.log(err);

}).Run();

}).catch(function (ex) {

console.log(ex);

});

}

function updateData(callback) {

var con = new msSqlConnecter.msSqlConnecter(config);

con.connect().then(function () {

new con.Request("update student set name = @name where id > @id")

.addParam("id", TYPES.Int, 3)

.addParam("name", TYPES.VarChar, "frank")

.onComplate(function (count) {

if (callback)

callback(count);

})

.onError(function (err) {

console.log(err);

})

.Run();

}).catch(function (ex) {

console.log(ex);

});

}

function deleteData(callback) {

var con = new msSqlConnecter.msSqlConnecter(config);

con.connect().then(function () {

new con.Request("delete from student where id > @id")

.addParam("id", TYPES.Int, 3)

.onComplate(function (count) {

if (callback)

callback(count);

})

.onError(function (err) {

console.log(err);

})

.Run();

}).catch(function (ex) {

console.log(ex);

});

}

console.log("inserting data");

insert(function (count) {

console.log("data inserted, listing data.");

queryAll(function (data) {

console.log(data);

console.log("updating data.");

updateData(function (count) {

console.log("data updated, listing data.");

queryAll(function (data) {

console.log(data);

console.log("deleting data.");

deleteData(function (count) {

console.log("data deleted,listing data.");

queryAll(function (data) {

console.log(data);

});

});

});

});

});

});

## More information

Document:

<http://tediousjs.github.io/tedious/>