MongoDB

Access data directly from your frontend code, intelligently distribute data for global apps, trigger serverless functions in response to data changes, and much more

Посилання

<https://www.mongodb.com/>

<https://mongodb.github.io/node-mongodb-native/?jmp=docs>

<https://metanit.com/web/nodejs/6.3.php>

<http://www.w3big.com/ru/mongodb/mongodb-dropdatabase.html>

Графічний клієнт для роботи з базою даних

<https://www.mongodb.com/download-center/compass>

Встановлення

npm install mongodb --save

За замовчуванням MongoDB в Windows використовує директорію C:\data\ . Якщо є потреба змінити розташування, то можна запустити процес із зазнеченням місця розташування

mongod --dbpath=/data

MongoDB відноситься до так званих NoSQL баз даних. На відміну від класичних SQL баз даних, дані зберігаються не у вигляді таблиць з записами, я у вигляді колекцій документів.



Кожен документ має унікальне значення (аналог первинного ключа \_\_id – не більше 128 символьно-цифрова послідовність), яке присвоюється кожному із документів (автоматично).

Загальна схема викристання

|  |  |
| --- | --- |
| Підключення модуля | const MongoClient = require("mongodb").MongoClient; |
| Створення об’єкта клієнта | const mongoClient = new MongoClient("<mongodb://localhost:27017/>", { useNewUrlParser: true }); |
| Відкриття з’єднання | mongoClient.connect |
| Підключення до бази даних | client.db("назва\_бд"); |
| Підключення до колекції | db.collection("назва\_колекції") |
| Маніпуляції з колекціями | collection.insertOne  collection. insertMany  collection. find  collection. findOne  collection. **deleteMany**  collection. **deleteOne**  collection.**findOneAndDelete**  collection.**drop**  . . . |
| Закриття з’єднання | client.close(); |

Приклад.

//--- Підключення модуля

const MongoClient = require("mongodb").MongoClient;

//--- Створення об’єкта клієнта

const url = "<mongodb://localhost:27017/>";

const mongoClient = new MongoClient(url, { useNewUrlParser: true });

 //--- Підлючення до сервера

mongoClient.connect(function(err, client){

    //--- Підключення до бази даних

    const db = client.db("usersdb");

//--- Підключення до колекції

    const collection = db.collection("users");

//--- Маніпуляції з колекціями

    let user = {name: "Tom", age: 23};

    collection.insertOne(user, function(err, result){ //---Додавання одного документу (запису)

        if(err){

            return console.log(err);

        }

        console.log(result.ops);

        client.close();

    });

});

Маніпуляції з базами даних

|  |  |  |
| --- | --- | --- |
| Під’єднання до бази даних | client.db("назва\_бд"); | **const** db = client.db(dbName); |
| Створення бази даних | створюється при додаванні першої колекції |  |
| Інф |  |  |
| Вилучення бази даних |  | db.dropDatabase() |
|  |  |  |

Маніпуляції з колекціями

|  |  |  |
| --- | --- | --- |
| Під’єднання до колекції | база\_даних .collection("назва\_колекції") | **const** col = db.collection(**'updates'**); |
| Створення колекції | створюється автоматично при вставці першого документа |  |
| Інф |  |  |
| Вилучення колекції |  | db.collection.drop() |
|  |  |  |

Маніпуляція з документами

|  |  |  |
| --- | --- | --- |
| Додавання |  |  |
| Вставка | колекція . insertOne ( документ, callback ) | db.collection(**'inserts'**).insertOne({a:1}, **function**(err, r) {. . .} |
| колекція . insertMany( масив\_документів , callback ) | db.collection(**'inserts'**).insertMany([{a:2}, {a:3}], **function**(err, r){. . .} |
| Пошук |  |  |
|  |  |
| Оновлення | колекція . updateOne | col.updateOne({a:1}, {$set: {b: 1}}, **function**(err, r) {. . .} |
| колекція . updateMany | col.updateMany({a:2}, {$set: {b: 1}}, **function**(err, r) {. . .} |
| Видалення | deleteOne | col.deleteOne({a:1}, **function**(err, r) {. . .} |
| deleteMany | col.deleteMany({a:2}, **function**(err, r) {. . .} |
| Пошук з операціями | колекція . **findOneAndUpdate** | col.findOneAndUpdate({a:1}, {$set: {b: 1}}, {  returnOriginal: **false**  , sort: [[a,1]]  , upsert: **true**  }, **function**(err, r) {. . .} |
|  | колекція . **findOneAndDelete** | col.findOneAndDelete({a:2}, **function**(err, r) {. . .} |
|  | колекція . **findOneAndReplace** |  |
| Виконання послідовності операцій | колекція . col.bulkWrite | col.bulkWrite([  { **insertOne:** { **document:** { **a:** 1 } } }  , { **updateOne:** { **filter:** {**a:**2}, **update:** {**$set:** {**a:**2}}, **upsert:true** } }  , { **updateMany:** { **filter:** {**a:**2}, **update:** {**$set:** {**a:**2}}, **upsert:true** } }  , { **deleteOne:** { **filter:** {**c:**1} } }  , { **deleteMany:** { **filter:** {**c:**1} } }  , { **replaceOne:** { **filter:** {**c:**3}, **replacement:** {**c:**4}, **upsert:true**}}]  , {**ordered:true**, **w:**1}, function(err, r) {  **assert**.equal(**null**, err);  **assert**.equal(1, r.insertedCount);  **assert**.equal(1, Object.keys(r.insertedIds).length);  **assert**.equal(1, r.matchedCount);  **assert**.equal(0, r.modifiedCount);  **assert**.equal(0, r.deletedCount);  **assert**.equal(2, r.upsertedCount);  **assert**.equal(2, Object.keys(r.upsertedIds).length);  *// Ordered bulk operation*  client.close();  }); |

<http://mongodb.github.io/node-mongodb-native/3.1/tutorials/crud/>

<https://nodejs.org/api/assert.html>

Групування

<http://mongodb.github.io/node-mongodb-native/3.1/tutorials/collations/>

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  | collection.aggregate(  [  { **'$group'**: { **'\_id'**: **"$first\_name"**, **'nameCount'**: { **'$sum'**: 1 } } },  { **'$sort'** : { **'\_id'** : 1 } }  ], { collation : { locale : **'de@collation=phonebook'** } },  **function**(err, docs) { |
|  |  |  |

Проекції