전북대 소중해유(You)

데이터베이스 연결 실습: MongoDB 활용

전북대학교 JEONBUK NATIONAL UNIVERSITY

전북대학교 SOFTWARE중심대학사업단

SW중심대학

과학기술정보통신부

IITP 정보통신기획평가원

01-MongoDB 연결 기본 예제

```javascript
const { MongoClient } = require('mongodb');

async function connectToDatabase() {
    const uri = "mongodb://localhost:27017";
    const client = new MongoClient(uri);

    try {
        await client.connect();
        console.log("Connected to MongoDB!");
    } catch (err) {
        console.error(err);
    } finally {
        await client.close();
    }
}

connectToDatabase();

```

02-데이터베이스 생성 및 컬렉션 생성

```
1   async function createDatabaseAndCollection() {
2       const uri = "mongodb://localhost:27017";
3       const client = new MongoClient(uri);
4
5       try {
6           await client.connect();
7           const db = client.db("myNewDatabase");
8           const collection = db.collection("myNewCollection");
9           console.log("Database and Collection created!");
10      } finally {
11          await client.close();
12      }
13  }
14
15  createDatabaseAndCollection();
16
```

03-문서 삽입 (InsertOne)

```
1  async function insertDocument() {
2      const uri = "mongodb://localhost:27017";
3      const client = new MongoClient(uri);
4
5      try {
6          await client.connect();
7          const db = client.db("myNewDatabase");
8          const collection = db.collection("myNewCollection");
9          const result = await collection.insertOne({ name: "Alice", age: 25, city: "New York" });
10         console.log(`Document inserted with _id: ${result.insertedId}`);
11     } finally {
12         await client.close();
13     }
14 }
15
16 insertDocument();
17
```

04-여러 문서 삽입 (InsertMany)

```
1  async function insertMultipleDocuments() {
2      const uri = "mongodb://localhost:27017";
3      const client = new MongoClient(uri);
4
5      try {
6          await client.connect();
7          const db = client.db("myNewDatabase");
8          const collection = db.collection("myNewCollection");
9          const documents = [
10             { name: "Bob", age: 30, city: "Los Angeles" },
11             { name: "Charlie", age: 28, city: "San Francisco" }
12         ];
13         const result = await collection.insertMany(documents);
14         console.log(`${result.insertedCount} documents inserted`);
15     } finally {
16         await client.close();
17     }
18 }
19
20 insertMultipleDocuments();
21
```

05-문서 조회 (FindOne)

```
1  async function findOneDocument() {
2      const uri = "mongodb://localhost:27017";
3      const client = new MongoClient(uri);
4
5      try {
6          await client.connect();
7          const db = client.db("myNewDatabase");
8          const collection = db.collection("myNewCollection");
9          const document = await collection.findOne({ name: "Alice" });
10         console.log("Found document:", document);
11     } finally {
12         await client.close();
13     }
14 }
15
16 findOneDocument();
17
```

06-여러 문서 조회 (Find)

```
1  async function findMultipleDocuments() {
2      const uri = "mongodb://localhost:27017";
3      const client = new MongoClient(uri);
4
5      try {
6          await client.connect();
7          const db = client.db("myNewDatabase");
8          const collection = db.collection("myNewCollection");
9          const documents = await collection.find({}).toArray();
10         console.log("Found documents:", documents);
11     } finally {
12         await client.close();
13     }
14 }
15
16 findMultipleDocuments();
17
```

07-문서 업데이트 (UpdateOne)

```
1  async function updateDocument() {
2      const uri = "mongodb://localhost:27017";
3      const client = new MongoClient(uri);
4
5      try {
6          await client.connect();
7          const db = client.db("myNewDatabase");
8          const collection = db.collection("myNewCollection");
9          const result = await collection.updateOne(
10             { name: "Alice" },
11             { $set: { city: "Boston" } }
12         );
13         console.log(`${result.matchedCount} document matched, ${result.modifiedCount} document updated`);
14     } finally {
15         await client.close();
16     }
17 }
18
19 updateDocument();
20
```

08-문서 삭제 (DeleteOne)

```
1   async function deleteDocument() {
2       const uri = "mongodb://localhost:27017";
3       const client = new MongoClient(uri);
4
5       try {
6           await client.connect();
7           const db = client.db("myNewDatabase");
8           const collection = db.collection("myNewCollection");
9           const result = await collection.deleteOne({ name: "Charlie" });
10          console.log(`${result.deletedCount} document deleted`);
11      } finally {
12          await client.close();
13      }
14  }
15
16  deleteDocument();
17
```

09-인덱스 생성

```
1  async function createIndex() {
2      const uri = "mongodb://localhost:27017";
3      const client = new MongoClient(uri);
4
5      try {
6          await client.connect();
7          const db = client.db("myNewDatabase");
8          const collection = db.collection("myNewCollection");
9          const result = await collection.createIndex({ name: 1 });
10         console.log(`Index created: ${result}`);
11     } finally {
12         await client.close();
13     }
14 }
15
16 createIndex();
17
```

10-트랜잭션 사용 예제

```javascript
async function runTransaction() {
    const uri = "mongodb://localhost:27017";
    const client = new MongoClient(uri);

    try {
        await client.connect();
        const session = client.startSession();
        session.startTransaction();

        const db = client.db("myNewDatabase");
        const collection = db.collection("myNewCollection");

        await collection.insertOne({ name: "Eve", age: 22, city: "Miami" }, { session });
        await collection.updateOne({ name: "Bob" }, { $set: { city: "Houston" } }, { session });

        await session.commitTransaction();
        console.log("Transaction committed successfully");
    } catch (err) {
        console.error("Transaction aborted due to an error:", err);
        await session.abortTransaction();
    } finally {
        await session.endSession();
        await client.close();
    }
}

runTransaction();

```

감사합니다.

- 본 온라인 콘텐츠는 2024년도 과학기술 정보통신부 및 정보통신기획평가원의 'SW중심대학사업' 지원을 받아 제작되었습니다.
- 본 결과물의 내용을 전재할 수 없으며, 인용(재사용)할 때에는 반드시 과학기술정보통신부와 정보통신기획평가원이 지원한 'SW중심대학'의 결과물이라는 출처를 밝혀야 합니다.

전북대학교 JEONBUK NATIONAL UNIVERSITY

전북대학교 SOFTWARE중심대학사업단

SW중심대학

과학기술정보통신부

IITP 정보통신기획평가원