To connect to MongoDB using Java, you will need to follow these steps:

1. \*\*Install MongoDB\*\*:

- Ensure that you have MongoDB installed and running on your machine. You can download and install MongoDB from the official website: [MongoDB Download Center](https://www.mongodb.com/try/download/community).

2. \*\*Add MongoDB Java Driver\*\*:

- You need the MongoDB Java driver to interact with MongoDB from your Java application.

- If you're using \*\*Maven\*\* for dependency management, add the following dependency to your `pom.xml`:

```xml

<dependency>

<groupId>org.mongodb</groupId>

<artifactId>mongo-java-driver</artifactId>

<version>4.8.0</version>

</dependency>

```

If you're not using Maven, you can download the driver JAR from the [official MongoDB website](https://mongodb.github.io/mongo-java-driver/) and add it to your project's classpath.

3. \*\*Create a Java Class to Connect to MongoDB\*\*:

Below is an example of a simple Java program to connect to a MongoDB instance and perform basic operations like inserting a document and querying it.

### Example Java Code:

```java

import com.mongodb.MongoClient;

import com.mongodb.MongoClientURI;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

import org.bson.Document;

public class MongoDBExample {

public static void main(String[] args) {

// 1. Connect to MongoDB

// MongoDB URI for localhost. If MongoDB is running locally with default port (27017)

String uri = "mongodb://localhost:27017"; // MongoDB URI (URL)

// Create a MongoClient object and connect to MongoDB

MongoClient mongoClient = new MongoClient(new MongoClientURI(uri));

// 2. Access Database

// In this example, we are accessing the "test" database

MongoDatabase database = mongoClient.getDatabase("test");

// 3. Access Collection

// "students" collection will hold documents

MongoCollection<Document> collection = database.getCollection("students");

// 4. Insert a Document

// Creating a document to represent a student

Document student = new Document("name", "John Doe")

.append("age", 21)

.append("department", "Computer Science")

.append("marks", 85);

// Insert the student document into the collection

collection.insertOne(student);

System.out.println("Document inserted successfully");

// 5. Query the Collection

// Retrieve all students in the "students" collection

for (Document doc : collection.find()) {

System.out.println(doc.toJson());

}

// 6. Close the connection

mongoClient.close();

}

}

```

### Explanation of Code:

1. \*\*Connect to MongoDB\*\*:

- The `MongoClient` object is used to connect to MongoDB. In this case, the MongoDB instance is assumed to be running on `localhost` with the default port `27017`.

- The `MongoClientURI` is used to create a connection string for MongoDB.

2. \*\*Access Database\*\*:

- The `mongoClient.getDatabase("test")` line accesses the database named `"test"`. If the database doesn't exist, MongoDB will create it when you insert data.

3. \*\*Access Collection\*\*:

- A collection is a group of documents within a database. In this case, we access a collection named `"students"`. If this collection doesn't exist, it will be created automatically.

4. \*\*Insert a Document\*\*:

- A `Document` represents a key-value pair in MongoDB, similar to a JSON object. We create a document to represent a student with fields like `name`, `age`, `department`, and `marks`.

- The `insertOne()` method inserts this document into the collection.

5. \*\*Query the Collection\*\*:

- The `find()` method retrieves all documents from the collection. We loop through all the documents and print them to the console.

6. \*\*Close the Connection\*\*:

- Once all operations are completed, we close the connection to MongoDB using `mongoClient.close()`.

### Important Points:

- \*\*MongoDB URI\*\*: The connection string (`uri`) used in `MongoClientURI(uri)` can be customized if you're connecting to a MongoDB instance that is running remotely, has authentication enabled, or uses a specific port.

For example:

```java

String uri = "mongodb://username:password@localhost:27017";

```

- \*\*Maven Dependency\*\*: If you're using Maven, the MongoDB driver dependency will manage the necessary JAR files for MongoDB.

### Running the Program:

- Ensure that MongoDB is running on your machine (`localhost:27017`).

- If you haven't installed MongoDB yet, you can start it by running the following command in the terminal (assuming MongoDB is installed):

```bash

mongod

```

- Now, run the Java program. It should connect to MongoDB, insert a student record, and print all student records from the `students` collection.

### Conclusion:

This simple example covers the basics of connecting to MongoDB from Java using the MongoDB Java Driver. It demonstrates how to insert and retrieve data from a MongoDB database, which is the foundation for more advanced operations you might need in your application.