Writing and Reading MySQL BLOB Using JDBC

Summary: this tutorial shows you how to write and read MySQL BLOB data using JDBC API.

We will use the `candidates` table in the mysqljdbc sample database. For the sake of demonstration, we will add one more column named `resume` into the `candidates` table. The data type of this column will be `MEDIUMBLOB` that can hold up to 16MB.

The following ALTER TABLE statement adds resume column to the `candidates` table.

```
ALTER TABLE candidates
ADD COLUMN resume LONGBLOB NULL AFTER email;
```

We will use a sample resume in PDF format and load this file into the `resume` column of the `candidates` table later. You can download the sample PDF file for practicing via the following link:

Download John Doe Resume in PDF format

Writing BLOB data into MySQL database

The steps for writing BLOB data into MySQL database is as follows:

First, open a new connection to the database by creating a new `Connection` object.

```
Connection conn = DriverManager.getConnection(url,username,password);
```

Then, construct an UPDATE statement and create a `PreparedStatement` from the `Connection` object.

```
String updateSQL = "UPDATE candidates "
                + "SET resume = ? "
                + "WHERE id=?";

PreparedStatement pstmt = conn.prepareStatement(updateSQL);
```

Next, read data from the sample resume file using `FileInputStream` and call `setBinarySt` `)` method to set parameters for the `PreparedStatement` .

After that, call the executeUpdate() method of the PreparedStatement object.

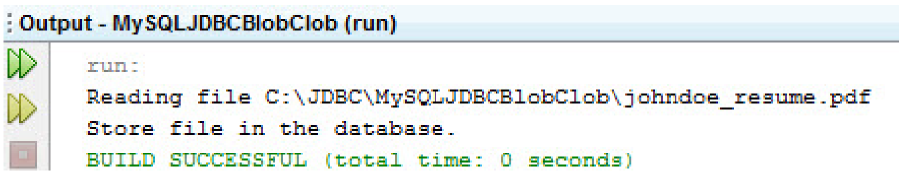
```
pstmt.executeUpdate();
```

Finally, close the PreparedStatement and Connection objects by calling the close() methods.

To simplify the Connection creation process, we use the MySQLJDBCUtil class that we developed in the previous tutorial to open a new connection. The complete example of writing BLOB data into MySQL database is as follows:

```
package org.mysqltutorial;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

/**
 *
 * @author mysqltutorial.org
 */
public class Main {

    /**
     * Update resume for a specific candidate
     *
     * @param candidateId
     * @param filename
     */
    public static void writeBlob(int candidateId, String filename) {
        // update sql
        String updateSQL = "UPDATE candidates "
                + "SET resume = ? "
                + "WHERE id=?";

        try (Connection conn = MySQLJDBCUtil.getConnection();
                PreparedStatement pstmt = conn.prepareStatement(updateSQL)) {

            // read the file
            File file = new File(filename);
            FileInputStream input = new FileInputStream(file);

            // set parameters
        } catch (SQLException | FileNotFoundException e) {
            System.out.println(e.getMessage());
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        writeBlob(122, "johndoe_resume.pdf");

    }

}
```

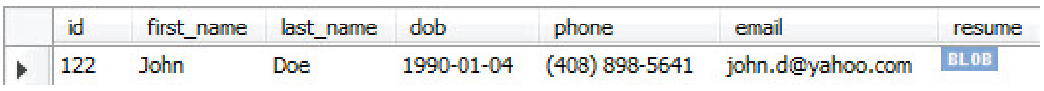
Let’s run the program.



```
Output - MySQLJDBCBlobClob (run)
run:
Reading file C:\JDBC\MySQLJDBCBlobClob\johndoe_resume.pdf
Store file in the database.
BUILD SUCCESSFUL (total time: 0 seconds)
```

Now we check the `candidates` table for the candidate with id 122.

```
SELECT * FROM candidates WHERE id = 122;
```



id	first_name	last_name	dob	phone	email	resume
122	John	Doe	1990-01-04	(408) 898-5641	john.d@yahoo.com	BLOB

As you see, we have BLOB data updated in the resume column of the `candidates` table for record with id 122.

Reading BLOB data from MySQL database

The process of reading BLOB data from the database is similar to the process of writing BLOB except for the part that we write BLOB data into the file.

First, open a new connection to the database.

```
PreparedStatement pstmt = conn.prepareStatement(selectSQL);
```

Next, set the parameters and execute the query:

```
pstmt.setInt(1, candidateId);
ResultSet rs = pstmt.executeQuery();
```

After that, get BLOB data from the `ResultSet` and write it into a file:

```
File file = new File(filename);
FileOutputStream output = new FileOutputStream(file);

System.out.println("Writing to file " + file.getAbsolutePath());
while (rs.next()) {
    InputStream input = rs.getBinaryStream("resume");
    byte[] buffer = new byte[1024];
    while (input.read(buffer) > 0) {
        output.write(buffer);
    }
}
```

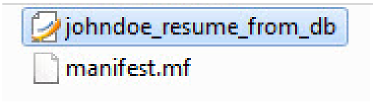
Finally, call the `close()` methods of `PreparedStatment` and `Connection` objects. If you use try-with-resources statement, you don't have to do it explicitly.

The following example illustrates how to read BLOB data from MySQL database.

```
package org.mysqltutorial;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * @author Main.org
 */
public class Main {

    /**
     * Read resume of a candidate and write it into a file
```

```
        ResultSet rs = null;

        try (Connection conn = MySQLJDBCUtil.getConnection();
                PreparedStatement pstmt = conn.prepareStatement(selectSQL);) {
            // set parameter;
            pstmt.setInt(1, candidateId);
            rs = pstmt.executeQuery();

            // write binary stream into file
            File file = new File(filename);
            FileOutputStream output = new FileOutputStream(file);

            System.out.println("Writing to file " + file.getAbsolutePath());
            while (rs.next()) {
                InputStream input = rs.getBinaryStream("resume");
                byte[] buffer = new byte[1024];
                while (input.read(buffer) > 0) {
                    output.write(buffer);
                }
            }
        } catch (SQLException | IOException e) {
            System.out.println(e.getMessage());
        } finally {
            try {
                if (rs != null) {
                    rs.close();
                }
            } catch (SQLException e) {
                System.out.println(e.getMessage());
            }
        }

    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        //
        readBlob(122, "johndoe_resume_from_db.pdf");
    }

}
```

After running the program, browsing the project the folder, you will see that there is a new file named `johndoe_resume_from_db.pdf` created.



In this tutorial, we have shown you how to work with MySQL BLOB data from JDBC.

Related Tutorials

- Introducing to JDBC
- Setting Up MySQL JDBC Development Environment
- Connecting to MySQL Using JDBC Driver
- Querying Data From MySQL Using JDBC
- Updating Data in MySQL Using JDBC PreparedStatement
- Inserting Data Into Table Using JDBC PreparedStatement
- MySQL JDBC Transaction
- Calling MySQL Stored Procedures from JDBC

Was this tutorial helpful ? Yes No

MYSQL QUICK START

What Is MySQL?

Install MySQL Database Server

Download MySQL Sample Database

Load Sample Database

MYSQL JDBC TUTORIAL

Introduction to JDBC

MySQL JDBC Setup

MySQL JDBC Connect

MySQL JDBC Select

MySQL JDBC PreparedStatement

MySQL JDBC Insert

MySQL JDBC Stored Procedures

MySQL JDBC Transaction

MySQL JDBC Read & Write BLOB

MYSQL PROGRAMMING INTERFACES

PHP MySQL Tutorial

RECENT MYSQL TUTORIALS

MySQL SHOW PROCESSLIST

MySQL JDBC Tutorial

MySQL INSERT INTO SELECT

MySQL ABS Function

MySQL MOD Function

MySQL ROLLUP

MySQL TRUNCATE Function

MySQL CEIL Function

OTHER TUTORIALS

MySQL Administration

MySQL Full-Text Search

MySQL Cheat Sheet

MySQL Books and Video Training

MySQL Hosting

MySQL Resources

ABOUT MYSQL TUTORIAL WEBSITE

MySQLTutorial.org is a website dedicated to MySQL database. We regularly publish useful MySQL tutorials to help web developers and database administrators learn MySQL faster and more effectively.

All MySQL tutorials are practical and easy-to-follow, with SQL script and screenshots available. More About Us

SITE LINKS

About Us

Contact Us

Request a Tutorial

Privacy Policy