Working with Large Objects (BLOB And CLOB)

Sometimes as the part of programming requirement,we have to insert and retrieve large files like images,video files,
audio files,resume etc wrt database.
Eg:upload image in matrinomial web sites
upload resume in job related web sites

To store and retrieve large information we should go for Large Objects(LOBs).

There are 2 types of Large Objects.
1. Binary Large Object (BLOB)
2. Character Large Object (CLOB)

1) Binary Large Object (BLOB)
A BLOB is a collection of binary data stored as a single entity in the database.
BLOB type objects can be images,video files,audio files etc..
BLOB datatype can store maximum of "4GB" binary data.

2) CLOB (Character Large Objects):
A CLOB is a collection of Character data stored as a single entity in the database.
CLOB can be used to store large text documents(may plain text or xml documents)
CLOB Type can store maximum of 4GB data.
Eg: resume.txt

Steps to insert BLOB type into database:
1. create a table in the database which can accept BLOB type data.
```
create table persons(name varchar2(10),image BLOB);
```
2. Represent image file in the form of Java File object.
```
File f = new File("sachin.jpg");
```
3. Create FileInputStream to read binary data represented by image file
```
FileInputStream fis = new FileInputStream(f)
```
4. Create PreparedStatement with insert query.
```
PreparedStatement pst = con.prepareStatement("insert into persons values(?,?)");
```
5. Set values to positional parameters.
```
pst.setString(1,"sachin");
```

To set values to BLOB datatype, we can use the following method: setBinaryStream()
```
public void setBinaryStream(int index,InputStream is)
public void setBinaryStream(int index,InputStream is,int length)
public void setBinaryStream(int index,InputStream is,long length)
```

6. execute sql query
```
pst.executeUpdate();
```

Steps to Retrieve BLOB type from Database

1. Prepare ResultSet object with BLOB type
```
ResultSet rs = st.executeQuery("select * from persons");
```

2. Read Normal data from ResultSet
```
String name=rs.getString(1);
```

3. Get InputStream to read binary data from ResultSet
```
InputStream is = rs.getBinaryStream(2);
```

4. Prepare target resource to hold BLOB data by using FileOutputStream

```
FileOutputStream fos = new FOS("katrina_new.jpg");
```

5. Read Binary Data from InputStream and write that Binary data to output Stream.

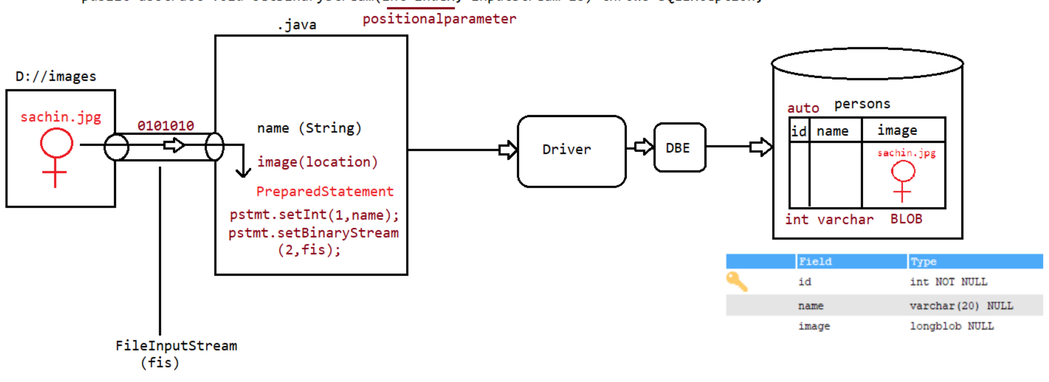
```
int i=is.read();
while(i!=-1)
{
      fos.write(i);
      is.read();
}

      or
byte[] b= new byte[2048];
while(is.read(b) > 0){
      fos.write(b);
  }
```

```
String sqlInsertQuery = "insert into persons(`name`,`image`) values (?,?)";
public abstract void setBinaryStream(int index, InputStream is) throws SQLException;
```

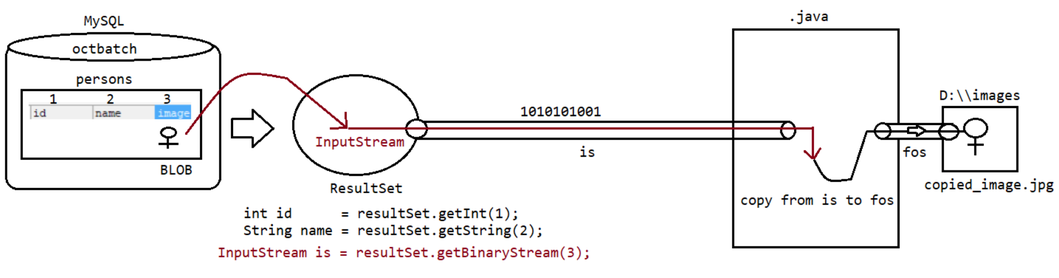
1,2

positionalparameter



	Field	Type
	id	int NOT NULL
	name	varchar(20) NULL
	image	longblob NULL

Retreival Operation



Performance low

```
int i = is.read();
while (i != -1) {
   fos.write(i);
   i = is.read();
}
```

Performance is high,but difficult for developers

```
byte[] b = new byte[1024];
while (is.read(b) > 0)
{
     fos.write(b);
}
```

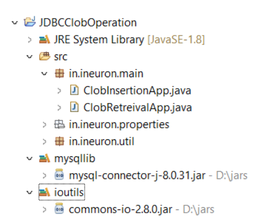
apache
commons-io.jar
`IOUtils.copy(is, fos);`

clob operation

Field	Type
id	int NOT NULL
name	varchar(20) NULL
history	longtext NULL

```
void setCharacterStream(int parameterIndex,Reader reader) throws SQLException;
```

```
pstmt.setCharacterStream(2, new FileReader(new File(pdfLoc)));;
```



Retrieval Operation

```
Reader reader = resultSet.getCharacterStream(3);
File file = new File("history_copied.txt");
FileWriter writer = new FileWriter(file);
IOUtils.copy(reader,writer);
```

```
SQLInjection
============
  Comments in SQL are of 2 types
    a. -- single line comment
    b. /*
          multiline comment
       */

eg: select count(*) from users where name='sachin' and password='tendulkar'; => no problem with inputs

    select count(*) from users where name='sachin' --' and password='tendulkar';
```

Statement Object
1. Query compiled('-- treated as comments)
2. Query executed('-- wont be in the execution phase)

comment

SQLInjection ⇒ end user manipulated the query with special syntax.
(Statement)

```
select count(*) from users where name= ? and password = ?
```

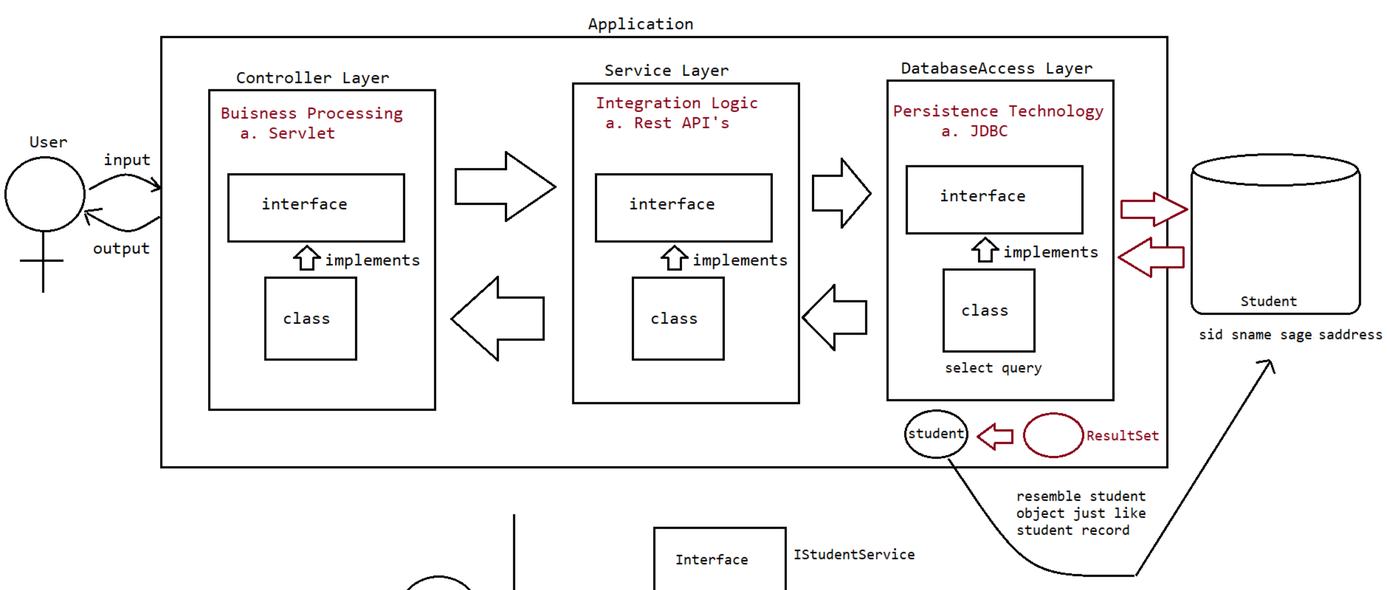
PreparedStatement Object
1. Query compiled ('-- wont come into picture)
2. Query executed

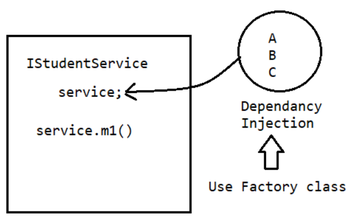
```
select count(*)
from users
where
   name = 'sachin'--
   and
   password= 'tendulkar'
```

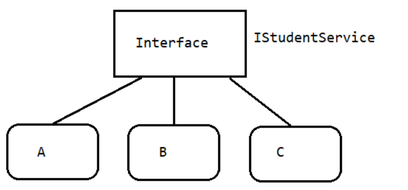
0

values

SQLInjection ⇒ end user manipulated the query with special syntax.
resolved through PrearedSatement







Body of methods available in implementaion clas
A class, B class, C class.

m1()	m1()	m1()
m2()	m2()	m2()
m3()	m3()	m3()