BLOB(byte 데이터) 처리

BLOB(byte 데이터) 처리

❖ 오라클 바이트 데이터 타입

- BLOB(Byte Large Object)

```
DROP TABLE GALLERY;

CREATE TABLE GALLERY (
    ID NUMBER PRIMARY KEY,
    TITLE VARCHAR2(256 CHAR),
    DESCRIPTION VARCHAR2(1024 CHAR),
    FILE_NAME VARCHAR2(256 CHAR),
    FILE_SIZE NUMBER,
    IMAGE BLOB,
    THUMB BLOB,
    REG_DATE DATE DEFAULT SYSDATE,
    UPDATE_DATE DATE DEFAULT SYSDATE
);

CREATE SEQUENCE GALLERY_SEQ;
```

BLOB(byte 데이터) 처리

❖ Gallery 모델

```
@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder

public class Gallery {
   private long id;
   private String title;
   private String description;
   private String fileName;
   private long fileSize;
   @ToString.Exclude private byte[] image;
   @ToString.Exclude private byte[] thumb;
   private Date regDate;
   private Date updateDate;
}
```

BLOB(byte 데이터) 처리

❖ GalleryDao

```
public interface GalleryDao
   extends CrudDao<Gallery, Long> ,
      PaginationDao<Gallery>,
      RandomDao<Gallery>{

}
```

BLOB(byte 데이터) 처리

❖ GalleryDaoImpl

```
public class GalleryDaoImpl extends
      PaginationDaoImpl<Gallery, Long>
      implements GalleryDao {

   public GalleryDaoImpl() {
      super("GalleryDao");
   }

   @Override
   public List<Gallery> random(int num) throws Exception {
      try (SqlSession session = Session.getSession()) {
         return session.selectList(
                     namespace + ".random", num);
      }
   }

}
```

BLOB(byte 데이터) 처리

❖ gallery-mapper.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
   PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
   "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="edu.iot.sagittarius2.dao.GalleryDao">

   <select id="count" resultType="int"><![CDATA[
      select count(*) from gallery
   ]]></select>

   <select id="getList" resultType="Gallery"><![CDATA[
      select
         id, title, description, file_name,
         reg_date, update_date
      from gallery
   ]]></select>
```

BLOB(byte 데이터) 처리

❖ gallery-mapper.xml

```
<select id="findById" parameterType="long"
        resultType="Gallery"><![CDATA[
   select * from gallery
   where id=#{id}
]]></select>

<insert id="insert" parameterType="Gallery"><![CDATA[
   insert into gallery (
      id, title, description, file_name, file_size, image, thumb
   )
   values(
      gallery_seq.nextval, #{title}, #{description},
      #{fileName}, #{fileSize}, #{image}, #{thumb}
   )
]]></insert>
```

BLOB(byte 데이터) 처리

❖ gallery-mapper.xml

```
<update id="update" parameterType="Gallery"><![CDATA[
   update gallery set
      title = #{title},
      description = #{title},
      update_date = sysdate
   where id=#{id}
]]></update>

<update id="changeImage" parameterType="Gallery"><![CDATA[
   update gallery set
      file_Name = #{fileName},
      file_size = #{fileSize},
      image = #{image},
      thumb = #{thumb},
      update_date = sysdate
   where id=#{id}
]]></update>
```

BLOB(byte 데이터) 처리

❖ gallery-mapper.xml

```
<delete id="delete" parameterType="long"><![CDATA[
   delete from gallery
   where id=#{id}
]]></delete>

<select id="getPage"   parameterType="map"
                resultType="Gallery"><![CDATA[
   select *
   from (
      select
         row_number() over (order by id desc) as seq,
         id, title, description, file_name,
         reg_date, update_date
      from gallery
   )
   where seq between #{start} and #{end}
]]></select>
```

BLOB(byte 데이터) 처리

❖ gallery-mapper.xml

```
   <select id="random" parameterType="int"
                  resultType="Gallery"><![CDATA[
      select * from(
         select * from gallery
         order by dbms_random.random
      ) where rownum <= #{num}
   ]]></select>
</mapper>
```

BLOB(byte 데이터) 처리

❖ 파일을 읽어서 byte[] 배열 얻기 – ByteArrayOutputStream 활용

```
public class FileUtil {
      :
   public static byte[] bytes(File file) throws Exception {

      ByteArrayOutputStream out = new ByteArrayOutputStream();
      try(BufferedInputStream bis = new BufferedInputStream(
               new FileInputStream(file)) ){

         int data ;
         while((data=bis.read())!=-1) {
            out.write(data);
         }
         return out.toByteArray();
      }
   }
```

BLOB(byte 데이터) 처리

❖ byte[] 배열을 파일에 저장하기

```
public class FileUtil {
      :
   public static void save(File file, byte[] data)throws Exception {
      try(
         FileOutputStream fos = new FileOutputStream(file)
      ){
         fos.write(data);
      }

   }
}
```

BLOB(byte 데이터) 처리

❖ 이미지 파일을 데이터베이스에 저장하기

```
public class GallerySaveEx1 {
   public static void main(String[] args) {
      File file = new File("c:/temp/Koala.jpg");

      try {
         byte[] data = FileUtil.bytes(file);
         String fname = file.getName();
         String title = fname.substring(0,
                                 fname.lastIndexOf("."));

         Gallery g = Gallery.builder()
                     .title(title)
                     .description(title)
                     .fileName(fname)
                     .fileSize(file.length())
                     .image(data)
                     .thumb(data)
                     .build();
```

BLOB(byte 데이터) 처리

❖ 이미지 파일을 데이터베이스에 저장하기

```
            // System.out.println(g);
            GalleryDao dao = new GalleryDaoImpl();
            dao.insert(g);

            System.out.println("저장완료");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

}
```

BLOB(byte 데이터) 처리

❖ 데이터베이스의 이미지를 파일로 저장하기(다운로드)

```
public class GalleryReadEx {
   public static void main(String[] args) {
      try {
         GalleryDao dao = new GalleryDaoImpl();
         Gallery g = dao.findById(1L);
         System.out.println(g);

         File dir = new File("c:/temp/download");
         if(!dir.exists()) {
            dir.mkdirs();
         }

         File imageFile = new File(dir, g.getFileName() );
         FileUtil.save(imageFile, g.getImage());

         File thumbFile = new File(dir, "thumb_" + g.getFileName() );
         FileUtil.save(thumbFile, g.getImage());
```

BLOB(byte 데이터) 처리

❖ 데이터베이스의 이미지를 파일로 저장하기(다운로드)

```
            System.out.println("OK");
         } catch (Exception e) {
            e.printStackTrace();
         }
      }

}
```