Java

java programs

- 동적 Jar loading 프로그램
- 폴더의 파일 열어 작업하기

동적 Jar loading 프로그램

동적 jar 로딩 및 동적 method 호출

- ```
  1  package com.exec;
  2
  3  import java.io.File;
  4  import java.io.FileFilter;
  5  import java.io.FilenameFilter;
  6  import java.lang.reflect.InvocationTargetException;
  7  import java.lang.reflect.Method;
  8  import java.net.URL;
  9  import java.net.URLClassLoader;
  10 import java.util.Arrays;
  11
  12 public class ExecTest {
  13
  14         public static void main(String[] args) throws NoSuchMethodException, S
  15
  16        // 코리안리 EAI customer function 위치 (고정 값)
  17                 String dir = "C:\\eclipse-workspace\\com.exec\\src\\main\\reso
  18        // 코리안리 EAI customer function prefix (고정 값)
  19                 String jarName = "kr.co.koreanre.eai";
  20        // 코리안리 EAI customer function Class 명 분류 할 수도 있다.
  21                 String className = "kr.co.koreanre.eai.execEaiFunctionClass";
  22        // 코리안리 EAI customer function method 명은 고정 예정
  23                 String methodName = "execMathod";
  24        // 코리안리 EAI customer function method에 전달되는 문자열 전달
  25                 String param = "Call method ,";
  26
  27                 ExecTest ex = new ExecTest();
  28
  29
  30                 // kr.co.koreanre.eai-202110280920.jar의 include 여부를 확인, inclu
  31                 if(!ex.checkLoadedJarList(dir,jarName))
  32                 {
  33                         System.out.println(" > Do not include : " + jarName );
  34            // 최신 버전으로 include
  35                         ex.loadJar(dir);
  36                 }
  37
  38                 String returnString = ex.execMethod(className, methodName, par
  39                 System.out.println(" > Return Value : " + returnString);
  40                 ex.checkLoadedJarList(dir, jarName);
  41         }
  42
  ```
  ```

```
43
44
45
46          /* ==========================================================================
47
48
49
50
51
52          /**
53           * kr.co.koreanre.eai-202110280920.jar의 로드 여부를 확인
54           *
55           * @param jarName
56           * @return boolean
57           */
58          private boolean checkLoadedJarList(String dir, String jarName) {
59                  System.out.println("\n>> Start Check loaded Jar list " );
60                  boolean includJar = false;
61
62                  ClassLoader classLoader = ClassLoader.getSystemClassLoader();
63                  URL[] urls = ((URLClassLoader)classLoader).getURLs();
64
65          for(URL url: urls){
66              if( url.getFile().toLowerCase().contains(".jar") ) {
67                  System.out.println(" > Included Jar : " + url.getFile());
68                      if(url.getFile().toLowerCase().contains(jarName))
69                      {
70                          if(isLoadNewVersionJar(dir, url.getFile()))
71                          {
72                                  includJar = true;
73                          }else {
74                                  includJar = false;
75                          }
76                      }else {
77                          includJar = false;
78                      }
79              }
80          }
81          return includJar;
82          }
83
84          /**
85      * 디렉토리 안에서 파라메터로 전달한 jar 파일이 최신인지 여부를 판단
86           * @param dir 검토 폴더
87           * @param jarName 비교 jar
88           * @return boolean
89           */
90          private boolean isLoadNewVersionJar(String dir, String jarName) {
91                  boolean isNweVersion = false;
92
```

```
 93                 File file = new File(dir);
 94                 File[] fileNameList = file.listFiles(new FilenameFilter() {
 95                         public boolean accept(File dir, String name) {
 96                                 return name.endsWith("jar");
 97                         }
 98                 });
 99
100                 if(fileNameList.length==0) {
101                         throw new NoClassDefFoundError("Koreanre EAI jar not e
102                 }else {
103                         Arrays.sort(fileNameList);
104                         String oldVersion = jarName.replace("\\", "/").substri
105                         String newVersion = fileNameList[fileNameList.length-1
106                         System.out.println(" > OLD : " + oldVersion );
107                         System.out.println(" > NEW : " + newVersion.substring(
108                         if(oldVersion.equals(newVersion.substring(newVersion.l
109                         {
110                                 System.out.println(" > is New Version");
111                                 isNweVersion = true;
112                         }else {
113                                 System.out.println(" > is Not New Version");
114                                 isNweVersion = false;
115                         }
116                 }
117
118                 return isNweVersion;
119         }
120
121         /**
122          * Method를 실행
123          * @param className
124          * @param methodName
125          * @param param
126          * @return
127          * @throws ClassNotFoundException
128          * @throws NoSuchMethodException
129          * @throws SecurityException
130          * @throws InstantiationException
131          * @throws IllegalAccessException
132          * @throws IllegalArgumentException
133          * @throws InvocationTargetException
134          */
135         private String execMethod(String className, String methodName, String
136                 System.out.println("\n>> Start exec method ! " );
137
138                 ClassLoader cl = Thread.currentThread().getContextClassLoader(
139                 Class<?> clazz = cl.loadClass(className);
140
141                 Method m = clazz.getMethod(methodName, param.getClass() );
142             m.setAccessible(true);
```

```
143
144             System.out.println(" > Classd Name : " + className);
145             System.out.println(" > Method Name : " + methodName);
146             System.out.println(" > Params Value : " + param);
147             System.out.println(" > Return Type : " + m.getReturnType());
148
149             if (m.getReturnType() != String.class) {
150                 System.out.println(" > Return type Stirng only : " + m.getRetu
151                 throw new NoSuchMethodException(methodName);
152             }
153
154             if (m.getModifiers() == 9) {
155                 System.out.println(" > Not allowed Static method : " + m.getMo
156                 throw new NoSuchMethodException(methodName);
157             }
158
159             if (m.getModifiers() != 1) {
160                 System.out.println(" > Public method only : " + m.getModifiers
161                 throw new NoSuchMethodException(methodName);
162             }
163
164            Object bbb = clazz.newInstance();
165             String returnMsg = "";
166         returnMsg = (String)m.invoke(bbb , new Object[] { param });
167
168             return returnMsg;
169
170         }
171
172
173         /**
174      * 해당 디렉토리의 최신 버전의 jar를 적용한다.
175          * @param dir
176          * @throws NoSuchMethodException
177          * @throws SecurityException
178          */
179         private void loadJar(final String dir) throws NoSuchMethodException, S
180                 System.out.println("\n>> LoadJar ");
181                 final URLClassLoader loader = (URLClassLoader)Thread.currentTh
182                 final Method method = URLClassLoader.class.getDeclaredMethod("
183                 method.setAccessible(true);
184
185                 new File(dir).listFiles(new FileFilter() {
186             public boolean accept(File jar) {
187                 if( jar.toString().toLowerCase().contains(".jar") ){
188                     try{
189                         if(isLoadNewVersionJar(dir, jar.toString())) {
190                                 method.invoke(loader, new Object[]{jar.toURI()
191                                 System.out.println(" > " + jar.toURI() + " is
192                         }
```

```
193
194                     }catch(Exception e){
195                         System.out.println(jar.toURI() + " can't load.");
196                     }
197                 }
198                                 return false;
199             }
200         });
201         }
202
203 }
```

- 호출을 받는 소스
  줄 번호 보이기/숨기기

```
1  package kr.co.koreanre.eai;
2
3  public class execEaiFunctionClass {
4
5          public execEaiFunctionClass() {
6                  System.out.println(" > Init #1" );
7          }
8
9          public String execMathod( String obj ) {
10
11                 String reString = new String();
12
13                 return reString + "= called test1";
14
15         }
16
17 }
```

- 호출을 받는 곳 pom.xml

```
1  <project xmlns="http://maven.apache.org/POM/4.0.0"
2          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.ap
4          <modelVersion>4.0.0</modelVersion>
5          <groupId>kr.co.koreanre.eai</groupId>
6          <artifactId>kr.co.koreanre.eai</artifactId>
7          <version>202110280920</version>
8          <name>Test1</name>
9
10         <build>
11                 <pluginManagement>
12                         <plugins>
13                                 <plugin>
14                                         <groupId>org.apache.maven.plugins</gro
15                                         <artifactId>maven-resources-plugin</ar
```

```
16                                      <version>2.3</version>
17                                  </plugin>
18                                  <plugin>
19                                      <artifactId>maven-resources-plugin</ar
20                                      <version>3.0.2</version>
21                                      <executions>
22                                          <execution>
23                                              <id>copy-resource-one<
24                                              <phase>install</phase>
25                                              <goals>
26                                                  <goal>copy-res
27                                              </goals>
28                                              <configuration>
29                                                  <outputDirecto
30                                                  <resources>
31                                                      <resou
32
33
34
35
36                                                      </reso
37                                                  </resources>
38                                              </configuration>
39                                          </execution>
40                                      </executions>
41                                  </plugin>
42                              </plugins>
43                          </pluginManagement>
44                  </build>
45
46  </project>
```

폴더의 파일 열어 작업하기

요건 :

특정 폴더의 파일을 모두 열러서 SQL에 있는 특정 문구로 시작하는 테이블명을 찾는다

쿼리당 하나 이상일 때는 하나로 카운트한다.

즉 쿼리하나에 하나의 API를 생성하여 타 도메인에서 현재 도메인의 테이블을 조회하는 것을 배제하여 DBMS를 분리할 수 있도록 한다.

Main 프로그램

```
package com.det;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class SocParsingWorker {

	public static String[] annotation_line = {"--", "//"};
	public static String[][] annotation_range = {

{	"/*", "*/", "/\\*", "\\*/"		}

,{	"<!--", "-->", "<!--", "-->"	}
                                                                        };
	public static String[][] searchKeyCategoryArry = {
                                                                { "<select", "</
select>" }
                                                                , { "<insert", "</
insert>" }
                                                                , { "<update", "</
update>" }
                                                                , { "<delete", "</
delete>" }
	};
	// 찾는 문자
	public static String findkeyString = "TB_FILEMANAGER_META";
```

```
        public static void main(String[] args) throws IOException {
                // TODO Auto-generated method stub
                System.out.println(" START ");
                String fileName = "";
                List<String> docRemoveAnn = new ArrayList<String>();
                StringRemovAnno stringRemovAnno = new StringRemovAnno();
                SearchString ss = new SearchString();
                Map<String, Integer> resultMapg = new HashMap<String, Integer>();  //
SocParsingWorker.searchKeyArry.length
                Map<String, Integer> totalResultMapg = new HashMap<String, Integer>();

                /* 파일 경로에 있는 파일 가져오기 */
                File rw = new File(args[0]);

                /* 파일 경로에 있는 파일 리스트 fileList[] 에 넣기 */
                File[] fileList = rw.listFiles();

                /* fileList에 있는거 for 문 돌려서 출력 */
                for (File file : fileList) {
                        if (file.isFile()) {
                                fileName = file.getName();
                                System.out.println("::::::::::::: PROC FILE NAME : " +
fileName + ":::::::::::::");

                                // Remove Annotation
                                // annotation_range
                                // annotation_line
                                docRemoveAnn =
stringRemovAnno.getStringRemovAnno(fileName, file.getPath());

                                // 주석 제거된 문서 docRemoveAnn를 활용
                                // searchKeyCategoryArry
                                resultMapg = ss.getSearchCnt(docRemoveAnn);
                                System.out.println("::::::::::::: File="+ fileName +" ,
Result=" + resultMapg + ":::::::::::::\n");

                                // Total add count by searchKeyCategoryArry
                                for (int i = 0; i <
SocParsingWorker.searchKeyCategoryArry.length; i++) {

totalResultMapg.put(SocParsingWorker.searchKeyCategoryArry[i][0].toUpperCase(),

toNlltoZero(totalResultMapg.get(SocParsingWorker.searchKeyCategoryArry[i]
[0].toUpperCase()))

+toNlltoZero(resultMapg.get(SocParsingWorker.searchKeyCategoryArry[i]
[0].toUpperCase()))
                                                                        );
                                }
```

```
                        }
                }
                System.out.println("\n\n\n>>>>>>>>>> All File Find [\""+ findkeyString
+"\"] Result=" + totalResultMapg);
        }

        private static int toNlltoZero(Integer integer) {
                if(integer==null) {
                        return 0;
                }
                return integer;
        }


}
```

주석 제거 클래스

범위 주석과 라인 주석을 제거

정규표현식으로 작성하다가 루프처리가 어려울 것 같고, /*, */는 했는데 <!-- -->는 안되어서 포기하고

정규표현식으로는 이하 생략과 이상 생략으로 정규 표현식으로 개발하고 나머지는 조건문으로 처리

태그가 불완전하면 많이 지워질 듯 함

```
package com.det;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;

public class StringRemovAnno {

        public List<String> getStringRemovAnno(String filename, String filepath)
throws IOException {
                System.out.println(" > fileName Path : " + filepath);
                List<String> fileLineArry = Files.readAllLines(Paths.get(filepath));
                List<String> removedAnnoRangeByte = fileLineArry;

                // 파일을 읽어서 범위의 주석을 삭제
                for (int i = 0; i < SocParsingWorker.annotation_range.length; i++) {
                        removedAnnoRangeByte = removedAnnoRangeByte(filename,
removedAnnoRangeByte, SocParsingWorker.annotation_range[i]);
                }
                // 라인을 읽어서 라인 주석을 삭제
                for (int i = 0; i < SocParsingWorker.annotation_line.length; i++) {
```

```
                        removedAnnoRangeByte = removedAnnoLineByte(filename,
removedAnnoRangeByte, SocParsingWorker.annotation_line[i]);
                }

                return removedAnnoRangeByte;
        }


        /**
         * 영역 주석 제거
         * @param filename
         * @param fileByte
         * @return
         */
        private List<String> removedAnnoRangeByte(String filename, List<String>
fileLineArry, String[] annRange) {
                System.out.println("  > Remove Anno Range : " + filename + " [" +
annRange[0] + " , " + annRange[1] + "]");
                List<String> removedAnnoRangeByte = new ArrayList<String>();

//              System.out.println("   > ORG One Line : " + fileLineArry);

                //for ann
                boolean annFlag = false;

                for (String oneLine : fileLineArry) {
                        int startAnnRag = oneLine.indexOf(annRange[0]);
                        int endAnnRag = oneLine.indexOf(annRange[1]);

//                      System.out.println("   > RED One Line : " + oneLine + "
---------->" + startAnnRag);

                        // 한줄에 모두 있을 때
                        if(startAnnRag >= 0 && endAnnRag >= 0){
                                if(!annFlag) {
                                        // 사이를 넣어라
                                        String ffoneLine =
oneLine.substring(0,startAnnRag);
                                        removedAnnoRangeByte.add(ffoneLine);
                                        oneLine =
oneLine.substring(endAnnRag+annRange[1].length(),oneLine.length());
                                        removedAnnoRangeByte.add(oneLine);
                                }

                        // 시작이 있을 때
                        }else if(startAnnRag >= 0) {
                                if(oneLine.trim().equals(annRange[0])) {
                                        // skip ... 이상하네 정규 표현식에서 /* 만 있을 때 /*가 표시
됨
                                }else {
```

```
                                        oneLine =
oneLine.replaceAll(""+annRange[2]+".*([^*]|[\\r\\n])", "");
                                        removedAnnoRangeByte.add(oneLine);
//                                      System.out.println("   > line : " + oneLine +
" [" + annRange[0] + " , " + annRange[1] + "] =========> START:" + startAnnRag );
                                }
                                annFlag = true;

                        // 끝이 있을 때
                        }else if(endAnnRag >= 0) {
                                oneLine = oneLine.replaceAll(".*"+annRange[3]+"*([^*]|
[\\r\\n])", "");
                                removedAnnoRangeByte.add(oneLine);
//                              System.out.println("   > line : " + oneLine + " [" +
annRange[0] + " , " + annRange[1] + "] =========> START:" + startAnnRag );
                                annFlag = false;
                        }

                        // 둘다 없고 시작이 안되었을 때
                        if(startAnnRag < 0 && endAnnRag < 0){
                                if(!annFlag) {
                                        removedAnnoRangeByte.add(oneLine);
                                }
                        }
                }
//              System.out.println("   > FLT One Line : " + removedAnnoRangeByte);

                //for ann
                return removedAnnoRangeByte;
        }

        /**
         * LIne 주석 제거
         * @param filename
         * @param removedAnnoRangeByte
         * @param string
         * @return
         */
        private List<String> removedAnnoLineByte(String filename, List<String>
fileLineArry, String ann) {
                System.out.println("  > Remove Anno Line : " + filename + " [" + ann +
"]");
                List<String> removedAnnoLineByte = new ArrayList<String>();
                for (String oneLine : fileLineArry) {
                        int startAnnLine = oneLine.indexOf(ann);
                        //System.out.println(".........>>>>>" + oneLine + " , " +
startAnnLine);
                        if( startAnnLine >= 0 ) {
                                oneLine = oneLine.substring(0,startAnnLine);
                                removedAnnoLineByte.add(oneLine);
```

```
                                //System.out.println("   > RED One Line : " +
oneLine);
                        }else {
                                removedAnnoLineByte.add(oneLine);
                        }
                }
                return removedAnnoLineByte;
        }

}
```

문구 찾기

파일을 하나의 긴 문자열로 만들고, 카타고리로 split 배열로 나누고 , 각 배열 문자열에 카테고리를 닫는 문구까지 잘라서 문자열을 만든다.

즉 <select로 자르면 다음 <select까지 나누어지기 때문에 </select>가 나오는 곳까지 잘라낸다.

그리고 검색 문구를 찾는다. 하나이상 나오면 카운트

```
package com.det;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class SearchString {

        public Map<String, Integer> getSearchCnt(List<String> docRemoveAnn) {
                StringBuffer fileToOneLine = new StringBuffer();
                Map<String, Integer> resultMapg = new HashMap<String, Integer>();

                for (String onLineNoAnno : docRemoveAnn) {
                        fileToOneLine.append(onLineNoAnno);
                }
//              System.out.println(fileToOneLine);

                // searchKeyArry 로 구분하여 배열로 정의
                for (int i = 0; i < SocParsingWorker.searchKeyCategoryArry.length; i+
+) {
//                      System.out.println("  >> SEARCH STAR ! key=\"" +
SocParsingWorker.searchKeyArry[i][0] +"\" ... \"" + SocParsingWorker.searchKeyArry[i]
[1] + "\"");
                        int searchCnt = 0;
                        String[] searchArry =
fileToOneLine.toString().toUpperCase().split(SocParsingWorker.searchKeyCategoryArry[i]
[0].toUpperCase());
                        for (int j = 0; j < searchArry.length; j++) {
```

```
                                String s = searchArry[j];
//              System.out.println("   > search : " + s);
                    if(s.indexOf(SocParsingWorker.searchKeyCategoryArry[i]
[1].toUpperCase()) >= 0) {
                        System.out.println("  >> Find! (" + j + ")" +
SocParsingWorker.searchKeyCategoryArry[i][0].toUpperCase() + "-" +
SocParsingWorker.searchKeyCategoryArry[i][1].toUpperCase()
                                            +" " +
s.indexOf(SocParsingWorker.searchKeyCategoryArry[i][1].toUpperCase()) + " Col");
                        s = s.substring(0,
s.indexOf(SocParsingWorker.searchKeyCategoryArry[i][1].toUpperCase()));
                        System.out.println("       \\_ TEXT = \"" + s.substring(0,50)
+ " ... \"");
                        if(s.indexOf(SocParsingWorker.findkeyString) >= 0) {
//                          System.out.println("   >> target String : " + s);
                            searchCnt++;
                        }
                    }
                }
                        System.out.println("   >> ::::::::: key=\"" +
SocParsingWorker.searchKeyCategoryArry[i][0] +"-" +
SocParsingWorker.searchKeyCategoryArry[i][1] + "\" ,
FIND(\""+SocParsingWorker.findkeyString+"\") Total Cnt = " + searchCnt);
                        resultMapg.put(SocParsingWorker.searchKeyCategoryArry[i]
[0].toUpperCase(), searchCnt);
                }

                return resultMapg;
        }

}
```

처리 파일

개발한 프로그램의 metaMapper.xml 파일과 임의 테스트용 파일을 처리함

metaMapper.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="com.joang.filemanager.meta.metaMapper">


<!-- ########## ============================================== COMMON START
============================================== ########## -->
<!-- ### For Paging ### -->
<sql id="pagingHeader">
```

```
SELECT A.* FROM (
SELECT row_number() over() AS rownumber, A.* FROM (
</sql>

<sql id="pagingFooter">
) A
) A WHERE A.rownumber BETWEEN #{start} AND #{last}
</sql>
<!-- ### For Paging ### -->
<!-- ########## ============================================== COMMON END
============================================== ########## -->




<!-- ########## ============================================== Meta List
Select ============================================== ########## -->

<select id="selectMetaListCnt"
parameterType="com.joang.filemanager.vo.MetaVO" resultType="int">
      <![CDATA[
SELECT COUNT( DISTINCT A.META ) AS total_pages
FROM TB_FILEMANAGER_META A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META_FILE C, TB_USERS_MAPPING D
WHERE 1=1
         AND A.IDX = B.META_IDX
         AND B.FILE_IDX = C.IDX
         AND C.IDX = D.FILE_IDX
   ]]>
AND D.USERNAME = #{userId}
AND A.DEL_GB = 'N'
AND C.DEL_GB = 'N'
<!-- 전체 -->
<if
test="positiveSearchKey != null and positiveSearchKey.size != 0 ">
<foreach collection="positiveSearchKey" item="item"
open="AND (" close=")" separator="AND">
A.META LIKE CONCAT('%',#{item},'%')
</foreach>
</if>
<if
test="negativeSearchKey != null and negativeSearchKey.size != 0 ">
AND C.IDX NOT IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="negativeSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
<if
test="andSearchKey != null and andSearchKey.size != 0 ">
AND C.IDX IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
```

```
WHERE C.IDX = B.META_IDX
<foreach collection="andSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
</select>

<select id="selectMetaList"
parameterType="com.joang.filemanager.vo.MetaVO"
resultType="com.joang.filemanager.vo.MetaVO">
<include refid="pagingHeader"></include>
      <![CDATA[
         /* com.joang.filemanager.meta.metaMapper.selectMetaList */
         SELECT
A.IDX,
A.META,
A.CREA_DTM,
A.CREA_ID,
COUNT( A.META ) AS COUNT_META
FROM
            TB_FILEMANAGER_META A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META_FILE C, TB_USERS_MAPPING D
         WHERE
            A.DEL_GB = 'N'
            AND C.DEL_GB = 'N'
         AND A.IDX = B.META_IDX
         AND B.FILE_IDX = C.IDX
         AND C.IDX = D.FILE_IDX
]]>
AND D.USERNAME = #{userId}

<!-- 전체 -->
<if
test="positiveSearchKey != null and positiveSearchKey.size != 0 ">
<foreach collection="positiveSearchKey" item="item"
open="AND (" close=")" separator="AND">
A.META LIKE CONCAT('%',#{item},'%')
</foreach>
</if>
<if
test="negativeSearchKey != null and negativeSearchKey.size != 0 ">
AND C.IDX NOT IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="negativeSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
   <if
test="andSearchKey != null and andSearchKey.size != 0 ">
AND C.IDX IN (
SELECT DISTINCT B.FILE_IDX
```

```
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="andSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
      <![CDATA[
         GROUP BY A.IDX,
A.META,
A.CREA_DTM,
A.CREA_ID
         ORDER BY A.IDX DESC
      ]]>
<include refid="pagingFooter"></include>
</select>


<select id="selectMetaAutocomplete"
parameterType="com.joang.filemanager.vo.MetaVO" resultType="map">
      <![CDATA[
         /* com.joang.filemanager.meta.metaMapper.selectMetaAutocomplete */
SELECT
A.IDX,
A.META,
A.FILE_CNT,
A.TO_CNT,
COUNT(E.FROM_IDX) AS FROM_CNT
FROM(
  SELECT A.IDX,
A.META,
A.FILE_CNT,
COUNT(E.FROM_IDX) AS TO_CNT
  FROM (
         SELECT
A.IDX,
A.META,
COUNT( A.META ) AS FILE_CNT
FROM
            TB_FILEMANAGER_META A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META_FILE C, TB_USERS_MAPPING D
         WHERE
            A.DEL_GB = 'N'
            AND C.DEL_GB = 'N'
         AND A.IDX = B.META_IDX
         AND B.FILE_IDX = C.IDX
         AND C.IDX = D.FILE_IDX
]]>
AND D.USERNAME = #{userId}
AND A.META LIKE '%'|| #{searchKey} || '%'

      <![CDATA[
      GROUP BY A.IDX, A.META
) A
left join TB_FILEMANAGER_META_REF E on  A.IDX = E.FROM_IDX
GROUP BY A.IDX, A.META, A.FILE_CNT, E.FROM_IDX
```

```
) A
left join TB_FILEMANAGER_META_REF E on  A.IDX = E.TO_IDX
GROUP BY A.IDX, A.META, A.FILE_CNT, A.TO_CNT, E.TO_IDX

          ORDER BY A.IDX DESC
      ]]>
</select>




<select id="selectUserAutocomplete"
parameterType="com.joang.common.vo.UserVO" resultType="map">
      <![CDATA[
         /* com.joang.filemanager.meta.metaMapper.selectUserAutocomplete */
         SELECT
tu.username
, tu.name
FROM tb_users tu
WHERE 1=1
AND ( tu.name LIKE '%'|| #{username} || '%' OR tu.username  LIKE '%'|| #{username} || '%' )
AND tu.del_gb = 'N'
      ]]>
</select>



<select id="selectRefMetaList"
parameterType="com.joang.filemanager.vo.MetaVO"
resultType="com.joang.filemanager.vo.RefMetaVO">
      <![CDATA[
/* com.joang.filemanager.meta.metaMapper.selectFromRefMetaList */
SELECT A.IDX, A.META, B.FROM_IDX, B.TO_IDX, B.META_REF
FROM TB_FILEMANAGER_META A,
(
SELECT TO_IDX AS IDX, FROM_IDX, B.TO_IDX, B.META_REF
FROM TB_FILEMANAGER_META A, TB_FILEMANAGER_META_REF B,
TB_FILEMANAGER_META_FILE C
WHERE
A.DEL_GB = 'N'
AND A.IDX = C.IDX
AND C.DEL_GB = 'N'
]]>

<!-- 전체 -->
<if
test="positiveSearchKey != null and positiveSearchKey.size != 0 ">
<foreach collection="positiveSearchKey" item="item"
open="AND (" close=")" separator="OR">
A.META LIKE CONCAT('%',#{item},'%')
</foreach>
</if>
<if
test="negativeSearchKey != null and negativeSearchKey.size != 0 ">
AND C.IDX NOT IN (
SELECT DISTINCT B.FILE_IDX
```

```
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="negativeSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
<if
test="andSearchKey != null and andSearchKey.size != 0 ">
AND C.IDX IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="andSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
      <![CDATA[
AND A.IDX = B.FROM_IDX
UNION
SELECT FROM_IDX AS IDX, FROM_IDX, B.TO_IDX, B.META_REF
FROM TB_FILEMANAGER_META A, TB_FILEMANAGER_META_REF B,
TB_FILEMANAGER_META_FILE C
WHERE
A.DEL_GB = 'N'
AND A.IDX = C.IDX
AND C.DEL_GB = 'N'

]]>

<!-- 전체 -->
<if
test="positiveSearchKey != null and positiveSearchKey.size != 0 ">
<foreach collection="positiveSearchKey" item="item"
open="AND (" close=")" separator="OR">
A.META LIKE CONCAT('%',#{item},'%')
</foreach>
</if>
<if
test="negativeSearchKey != null and negativeSearchKey.size != 0 ">
AND C.IDX NOT IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="negativeSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
<if
test="andSearchKey != null and andSearchKey.size != 0 ">
AND C.IDX IN (
SELECT DISTINCT B.FILE_IDX
```

```
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="andSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
      <![CDATA[
AND A.IDX = B.TO_IDX
) B
WHERE A.IDX = B.IDX
      ]]>
</select>

<select id="selectMetaValueByMetaIdx" parameterType="int"
resultType="string">
SELECT META FROM TB_FILEMANAGER_META
WHERE IDX = #{meta_idx}
</select>


<!-- ########## ============================================== Meta List
Select ============================================== ########## -->




<!-- ########## ============================================== File List/Detail
Select ============================================== ########## -->

<select id="selectFileListCnt"
parameterType="com.joang.filemanager.vo.FileVO" resultType="int">
      <![CDATA[
SELECT COUNT( DISTINCT A.IDX ) AS TOTAL_PAGES
]]>

<choose>
<when test="searchKey != null and searchKey != '' ">
FROM TB_FILEMANAGER_META_FILE A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META C, TB_USERS_MAPPING D
</when>
<otherwise>
FROM
TB_FILEMANAGER_META_FILE A, TB_USERS_MAPPING D
</otherwise>
</choose>
WHERE
A.DEL_GB ='N'
AND A.IDX = D.FILE_IDX
AND D.USERNAME = #{creaId}

<choose>
<when test="searchKey != null and searchKey != '' ">
AND B.FILE_IDX = A.IDX
AND C.IDX = B.META_IDX
```

```
</when>
</choose>
<!-- 전체 -->
<if
test="positiveSearchKey != null and positiveSearchKey.size != 0 ">
<foreach collection="positiveSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
</if>
<if
test="negativeSearchKey != null and negativeSearchKey.size != 0 ">
AND A.IDX NOT IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="negativeSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
<if
test="andSearchKey != null and andSearchKey.size != 0 ">
AND A.IDX IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="andSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
</select>

<select id="selectFileList"
parameterType="com.joang.filemanager.vo.FileVO"
resultType="com.joang.filemanager.vo.FileVO">
<include refid="pagingHeader"></include>
      <![CDATA[
      /* com.joang.filemanager.meta.metaMapper.selectFileList */
         SELECT
A.IDX,
E.OPN_CNT,
A.FILE_TITLE,
A.FILE_TYPE,
SUBSTRING(A.FILE_DETAIL::varchar,0,100) || '...' AS FILE_DETAIL,
A.ORIGINAL_FILE_NAME,
A.STORED_FILE_NAME,
A.FILE_SIZE,
A.CREA_DTM,
A.CREA_ID,
A.DEL_GB,
A.DOWN_YN,
A.STAR_LEVEL
      ]]>
```

```
<choose>
<when test="searchKey != null and searchKey != '' ">
, C.META AS SEARCH_KEY
FROM
TB_FILEMANAGER_META_FILE A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META C, TB_USERS_MAPPING D
</when>
<otherwise>
FROM
TB_FILEMANAGER_META_FILE A, TB_USERS_MAPPING D
</otherwise>
</choose>
, (SELECT COUNT(IDX) AS OPN_CNT, FILE_IDX  FROM TB_USERS_MAPPING TUM GROUP BY
FILE_IDX ) E

      <![CDATA[
         WHERE
            A.DEL_GB ='N'
            AND A.IDX = D.FILE_IDX
            AND A.IDX = E.FILE_IDX
      ]]>
AND D.USERNAME = #{creaId}

<choose>
<when test="searchKey != null and searchKey != '' ">
AND B.FILE_IDX = A.IDX
AND C.IDX = B.META_IDX
</when>
</choose>
<!-- 전체 -->
<if
test="positiveSearchKey != null and positiveSearchKey.size != 0 ">
<foreach collection="positiveSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
</if>
<if
test="negativeSearchKey != null and negativeSearchKey.size != 0 ">
AND A.IDX NOT IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="negativeSearchKey" item="item"
open="AND (" close=")" separator="OR">
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
<if
test="andSearchKey != null and andSearchKey.size != 0 ">
AND A.IDX IN (
SELECT DISTINCT B.FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING B, TB_FILEMANAGER_META C
WHERE C.IDX = B.META_IDX
<foreach collection="andSearchKey" item="item"
open="AND (" close=")" separator="OR">
```

```
C.META LIKE CONCAT('%',#{item},'%')
</foreach>
)
</if>
<choose>
<when test="searchKey != null and searchKey != '' ">
<![CDATA[
ORDER BY CONCAT(A.STAR_LEVEL, A.IDX)
]]>
</when>
<otherwise>
	<![CDATA[
ORDER BY A.IDX
	]]>
</otherwise>
</choose>
<choose>
<when test="orderBy == 'ASC'">
	<![CDATA[
ASC
	]]>
</when>
<otherwise>
	<![CDATA[
DESC
	]]>
</otherwise>
</choose>
<include refid="pagingFooter"></include>
</select>


<!-- file detail -->
<select id="selectFileDetail" parameterType="int"
resultType="com.joang.filemanager.vo.FileVO">
	<![CDATA[
	/* com.joang.filemanager.meta.metaMapper.selectFileDetail */
	   SELECT
IDX,
E.OPN_CNT,
FILE_TITLE,
FILE_TYPE,
FILE_DETAIL,
ORIGINAL_FILE_NAME,
STORED_FILE_NAME,
FILE_SIZE,
CREA_DTM,
CREA_ID,
DEL_GB,
DOWN_YN,
ERROR,
STAR_LEVEL
FROM TB_FILEMANAGER_META_FILE, (SELECT COUNT(IDX) AS OPN_CNT, FILE_IDX  FROM
TB_USERS_MAPPING TUM GROUP BY FILE_IDX ) E
WHERE IDX = #{idx}
AND E.FILE_IDX = #{idx}
AND DEL_GB = 'N'
```

```
    ]]>
</select>

<select id="selectFileShareUserListByFile" parameterType="int"
resultType="com.joang.filemanager.vo.FileShareUserVO">
    <![CDATA[
      SELECT
B.USERNAME,
C.NAME,
B.FILE_IDX,
B.CREA_DTM,
B.CREA_ID
FROM TB_FILEMANAGER_META_FILE A, TB_USERS_MAPPING B, TB_USERS C
WHERE 1=1
AND A.IDX = #{idx}
AND A.IDX = B.FILE_IDX
AND C.USERNAME = B.USERNAME
    ]]>
</select>

<select id="getTargetMetaIdx" resultType="string">
SELECT COALESCE(MAX(
IDX ), 0) FROM TB_FILEMANAGER_META_FILE
</select>

<!-- ########## ============================================== File List/Detail
Select ============================================== ########## -->




<!-- ########## ============================================== Meta INSERT/UPDATE
Select ============================================== ########## -->

<select id="checkMeta" parameterType="String"
resultType="hashmap">
<![CDATA[
      SELECT
idx
FROM
        TB_FILEMANAGER_META
      WHERE
        DEL_GB ='N'
AND TRIM(META) = TRIM(#{meta})
    ]]>
</select>

<insert id="insertMeta" parameterType="hashmap">
  <![CDATA[
    INSERT INTO TB_FILEMANAGER_META
    (
      USER_ID, META, CREA_DTM, CREA_ID, DEL_GB
    )
    VALUES
    (
      #{userId},
      #{meta},
```

```
        NOW(),
        #{userId},
        'N'
      )
  ]]>
<selectKey keyProperty="meta_idx" resultType="int"
order="AFTER">
SELECT COALESCE(MAX( IDX ), 0) AS META_IDX FROM TB_FILEMANAGER_META
</selectKey>
</insert>

<insert id="insertMetaMapping" parameterType="hashmap">
<selectKey keyProperty="idx" resultType="int" order="BEFORE">
SELECT COALESCE(MAX( IDX ), 0) FROM TB_FILEMANAGER_META_FILE
</selectKey>
  <![CDATA[
    INSERT INTO TB_FILEMANAGER_META_MAPPING
    (
      USER_ID, META_IDX, FILE_IDX, CREA_DTM, CREA_ID, DEL_GB
    )
    VALUES
    (
      #{userId},
      #{meta_idx}::INTEGER,
  ]]>
<choose>
<when test="file_idx != null">
<!-- 변경되어야할 파일의 file_idx -->
#{file_idx}::INTEGER,
</when>
<otherwise>
<!-- 새 파일 업로드인 경우 idx -->
#{idx}::INTEGER,
</otherwise>
</choose>
  <![CDATA[
        NOW(),
        #{userId},
        'N'
      )
  ]]>
</insert>

<insert id="insertMetaExistMapping" parameterType="hashmap">
  <![CDATA[
    INSERT INTO TB_FILEMANAGER_META_MAPPING
    (
      USER_ID, META_IDX, FILE_IDX, CREA_DTM, CREA_ID, DEL_GB
    )
    VALUES
    (
      #{userId},
      #{meta_idx}::INTEGER,
      #{idx}::INTEGER,
      NOW(),
      #{userId},
      'N'
```

```
      )
    ]]>
</insert>


<insert id="insertShareUser" parameterType="com.joang.filemanager.vo.FileShareUserVO">
    <![CDATA[
      INSERT INTO TB_USERS_MAPPING
      (
        USERNAME, FILE_IDX, CREA_DTM, CREA_ID
      )
      VALUES
      (
        #{username},
        #{file_idx}::INTEGER,
        NOW(),
        #{crea_id}
      )
    ]]>
</insert>

<!-- Meta Merge -->
<insert id="updateMergeMetaMapping" parameterType="hashmap">
/*
updateMergeMetaMapping */
UPDATE TB_FILEMANAGER_META_MAPPING SET
META_IDX = #{targetMeta}::INTEGER WHERE META_IDX =
#{sourceMeta}::INTEGER;
</insert>

<insert id="updateMergeMeta" parameterType="hashmap">
UPDATE
TB_FILEMANAGER_META SET
DEL_GB = 'Y' WHERE IDX = #{sourceMeta}::INTEGER;
</insert>

<!-- Delete Meta Detail -->
<insert id="deleteMetaDetail" parameterType="int">
DELETE FROM
TB_FILEMANAGER_META
WHERE IDX = #{meta_idx};
</insert>


<!-- ########## ============================================== Meta INSERT/UPDATE
Select ============================================== ########## -->




<!-- ########## ============================================== File INSERT/UPDATE
Select ============================================== ########## -->

<insert id="insertFile" parameterType="hashmap">
    <![CDATA[
      INSERT INTO TB_FILEMANAGER_META_FILE
```

```
      (
        FILE_TITLE,
        FILE_DETAIL,
        ORIGINAL_FILE_NAME,
        STORED_FILE_NAME,
        FILE_SIZE,
        FILE_TYPE,
        CREA_DTM,
        CREA_ID,
        DOWN_YN,
        STAR_LEVEL
      )
      VALUES
      (
        #{file_title},
        #{file_detail},
        #{original_file_name},
        #{stored_file_name},
        #{file_size},
        #{file_type},
        NOW(),
        #{crea_id},
        #{down_yn},
        #{star_level}::INTEGER
      )
    ]]>
</insert>

<insert id="updateFile" parameterType="hashmap">
    <![CDATA[
UPDATE  TB_FILEMANAGER_META_FILE SET
      ]]>
<if test="file_title != null and file_title != ''">
FILE_TITLE = #{file_title} ,
</if>
<if test="file_type != null and file_type != ''">
FILE_TYPE = #{file_type} ,
</if>
<if test="file_detail != null and file_detail != ''">
FILE_DETAIL = #{file_detail} ,
</if>
<if test="crea_id != null and crea_id != ''">
CREA_ID = #{crea_id} ,
</if>
<if test="star_level != null and star_level != ''">
STAR_LEVEL = #{star_level}::INTEGER ,
</if>
      <![CDATA[
   CREA_DTM = NOW()
WHERE IDX = #{idx}::INTEGER;
   ]]>
</insert>

<delete id="deleteShareUsers" parameterType="int">
    <![CDATA[
DELETE FROM TB_USERS_MAPPING
WHERE FILE_IDX = #{file_idx}
```

```
]]>
</delete>

<select id="selectMetaListByFile" parameterType="int"
resultType="com.joang.filemanager.vo.MetaVO">
      <![CDATA[
        SELECT
A.IDX,
A.USER_ID,
A.META,
A.CREA_DTM,
A.CREA_ID,
A.DEL_GB
FROM TB_FILEMANAGER_META A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META_FILE C
WHERE A.IDX = B.META_IDX
AND B.FILE_IDX = C.IDX
AND C.DEL_GB = 'N'
AND C.IDX = #{idx}
      ]]>
</select>


<!-- update File Detail -->
<insert id="updateFileDetail" parameterType="hashmap">
UPDATE
TB_FILEMANAGER_META_FILE SET
DEL_GB = 'Y' WHERE IDX = #{idx}::INTEGER;
</insert>

<!-- File Mapping update -->
<insert id="updateFileMetaMapping" parameterType="hashmap">
UPDATE
TB_FILEMANAGER_META_MAPPING SET
FILE_IDX = #{targetMeta}::INTEGER WHERE FILE_IDX = #{sourceMeta}::INTEGER;
</insert>

<!-- Delete File detail -->
<delete id="deleteFileDetail" parameterType="int">
<![CDATA[
        DELETE
FROM
          TB_FILEMANAGER_META_FILE
        WHERE
          IDX = #{file_idx}::INTEGER;
    ]]>
</delete>

<update id="updateYouTuBeDownloadYn" parameterType="int">
/*
updateYouTuBeDownloadYn */
UPDATE TB_FILEMANAGER_META_FILE SET
DOWN_YN = 'N'
WHERE
IDX = #{file_idx};
</update>
```

```
<!-- ########## ============================================== File INSERT/UPDATE
Select ============================================== ########## -->




<!-- ########## ============================================== MApping Management
============================================== ########## -->

<!-- ########## Meta Ref ########## -->
<select id="selectMetaRefList"
parameterType="com.joang.filemanager.vo.MetaRefVO"
resultType="com.joang.filemanager.vo.MetaRefVO">
      <![CDATA[
         SELECT
A.FROM_IDX,
A.TO_IDX,
A.META_REF,
A.CREA_DTM,
A.CREA_ID
FROM TB_FILEMANAGER_META_REF A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META_FILE C, TB_USERS_MAPPING D
WHERE 1=1
      ]]>
<if test="fromIdx != null and fromIdx != '' and fromIdx != 0">
<![CDATA[
AND A.FROM_IDX = #{fromIdx}
]]>
</if>
<if test="toIdx != null and toIdx != '' and toIdx != 0">
<![CDATA[
AND A.TO_IDX = #{toIdx}
]]>
</if>
      <![CDATA[
         AND A.FROM_IDX = B.META_IDX
         AND B.FILE_IDX = C.IDX
AND A.DEL_GB = 'N'
AND C.IDX = D.FILE_IDX
AND D.USERNAME = #{crea_id}
UNION
         SELECT
A.FROM_IDX,
A.TO_IDX,
A.META_REF,
A.CREA_DTM,
A.CREA_ID
FROM TB_FILEMANAGER_META_REF A, TB_FILEMANAGER_META_MAPPING B,
TB_FILEMANAGER_META_FILE C, TB_USERS_MAPPING D
WHERE 1=1
      ]]>
<if test="fromIdx != null and fromIdx != '' and fromIdx != 0">
<![CDATA[
AND A.FROM_IDX = #{fromIdx}
]]>
</if>
```

```
<if test="toIdx != null and toIdx != '' and toIdx != 0">
<![CDATA[
AND A.TO_IDX = #{toIdx}
]]>
</if>
      <![CDATA[
         AND A.TO_IDX = B.META_IDX
         AND B.FILE_IDX = C.IDX
AND A.DEL_GB = 'N'
AND C.IDX = D.FILE_IDX
AND D.USERNAME = #{crea_id}
      ]]>
</select>

<!-- Check mapping -->
<select id="checkMetaMapping" parameterType="hashmap"
resultType="hashmap">
<![CDATA[
         SELECT
meta_idx
FROM
            TB_FILEMANAGER_META_MAPPING
         WHERE
            DEL_GB ='N'
            AND META_IDX = #{meta_idx}::INTEGER
AND FILE_IDX = #{file_idx}::INTEGER
      ]]>
</select>


<!-- Delete meta mapping -->
<delete id="deleteMetaMapping" parameterType="hashmap">
<![CDATA[
         DELETE
FROM
            TB_FILEMANAGER_META_MAPPING
         WHERE
            META_IDX =#{meta_idx}::INTEGER
AND FILE_IDX = #{file_idx}::INTEGER
      ]]>
</delete>

<select id="selectMappingListByMetaIdx" parameterType="int"
resultType="com.joang.filemanager.vo.MetaVO">
      <![CDATA[
         SELECT
IDX,
META_IDX,
FILE_IDX
FROM TB_FILEMANAGER_META_MAPPING
WHERE META_IDX = #{idx}
      ]]>
</select>

<!-- Delete Mapping -->
<delete id="deleteMetaMappingByFileIdx" parameterType="int">
<![CDATA[
```

```
        DELETE
FROM
          TB_FILEMANAGER_META_MAPPING
        WHERE
          FILE_IDX = #{file_idx}
    ]]>
</delete>

<insert id="insertMetaRef"
parameterType="com.joang.filemanager.vo.MetaRefVO">
<![CDATA[
INSERT INTO TB_FILEMANAGER_META_REF (FROM_IDX, TO_IDX, META_REF, CREA_DTM,
CREA_ID, UPDT_DTM, DEL_GB) VALUES
(
#{fromIdx}, #{toIdx}, #{metaRef}, NOW(), #{creaId}, NOW(), 'N'
)
]]>
</insert>


<insert id="updateMetaRef"
parameterType="com.joang.filemanager.vo.MetaRefVO">
<![CDATA[
UPDATE  TB_FILEMANAGER_META_REF SET
META_REF =  #{metaRef},
UPDT_DTM = NOW()
WHERE
FROM_IDX = #{fromIdx}
AND TO_IDX = #{toIdx}
AND CREA_ID = #{crea_id}
]]>
</insert>


<insert id="deleteMetaRef"
parameterType="com.joang.filemanager.vo.MetaRefVO">
<![CDATA[
DELETE FROM TB_FILEMANAGER_META_REF
WHERE
FROM_IDX = #{fromIdx}
AND TO_IDX = #{toIdx}
AND CREA_ID = #{crea_id}
]]>
</insert>


<update id="updateMergeMetaFromRef" parameterType="hashmap">
/*
updateMergeMetaRef */
UPDATE TB_FILEMANAGER_META_REF SET
FROM_IDX = #{targetFromMeta}::INTEGER
WHERE
FROM_IDX = #{sourceFromMeta}::INTEGER AND TO_IDX = #{sourceToMeta}::INTEGER
AND CREA_ID = #{crea_id}
</update>

<update id="updateMergeMetaToRef" parameterType="hashmap">
```

```
/*
updateMergeMetaRef */
UPDATE TB_FILEMANAGER_META_REF SET
TO_IDX = #{targetToMeta}::INTEGER
WHERE
TO_IDX = #{sourceToMeta}::INTEGER AND FROM_IDX = #{sourceFromMeta}::INTEGER
AND CREA_ID = #{crea_id}
</update>

<delete id="daleteMergeMetaRef" parameterType="hashmap">
/*
daleteMergeMetaFromRef */
DELETE
FROM
TB_FILEMANAGER_META_REF
WHERE
FROM_IDX = #{sourceFromMeta}::INTEGER AND TO_IDX = #{sourceToMeta}::INTEGER
AND CREA_ID = #{crea_id}
</delete>

<delete id="deleteMetaMappingFROMRef" parameterType="int">
/*
deleteMetaMappingFROMRef */
DELETE
FROM
TB_FILEMANAGER_META_REF
WHERE
FROM_IDX = #{deleteMeta}
</delete>

<delete id="deleteMetaMappingTORef" parameterType="int">
/*
deleteMetaMappingTORef */
DELETE
FROM
TB_FILEMANAGER_META_REF
WHERE
TO_IDX = #{deleteMeta}
</delete>


<!-- ########## ============================================== Mapping Management
============================================== ########## -->


</mapper>
```

test.txt

```
test1
test2 // hyunsu
hhhh /* hello 0

hello

mmmm */ llll
```

```
/*
여기
*/

123456 <!-- test
   hdewhfd
   wehfiuheri
  --> dfjrwevniearlivsefh

1234 <!-- hhyubn --> 555555555555
hhhh -- nfejeewfhlwjka

test /* hahahah */
<!--
fwef -->
우리 대한민구 만세 // 류재환
FINE
```

처리 결과

처리결과

```
START
:::::::::::: PROC FILE NAME : metaMapper.xml::::::::::::
> fileName Path : C:\HYUNSU\eclipse-showcase-
workspace\SocParsing\parsing\metaMapper.xml
> Remove Anno Range : metaMapper.xml [/* , */]
> Remove Anno Range : metaMapper.xml [<!-- , -->]
> Remove Anno Line : metaMapper.xml [--]
> Remove Anno Line : metaMapper.xml [//]
>> Find! (1)<SELECT-</SELECT> 1415 Col
\_ TEXT = " ID="SELECTMETALISTCNT" PARAMETERTYPE="COM.JOANG. ... "
>> Find! (2)<SELECT-</SELECT> 1760 Col
\_ TEXT = " ID="SELECTMETALIST" PARAMETERTYPE="COM.JOANG.FIL ... "
>> Find! (3)<SELECT-</SELECT> 1199 Col
\_ TEXT = " ID="SELECTMETAAUTOCOMPLETE" PARAMETERTYPE="COM.J ... "
>> Find! (4)<SELECT-</SELECT> 331 Col
\_ TEXT = " ID="SELECTUSERAUTOCOMPLETE" PARAMETERTYPE="COM.J ... "
>> Find! (5)<SELECT-</SELECT> 2723 Col
\_ TEXT = " ID="SELECTREFMETALIST" PARAMETERTYPE="COM.JOANG. ... "
>> Find! (6)<SELECT-</SELECT> 136 Col
\_ TEXT = " ID="SELECTMETAVALUEBYMETAIDX" PARAMETERTYPE="INT" ... "
>> Find! (7)<SELECT-</SELECT> 1610 Col
\_ TEXT = " ID="SELECTFILELISTCNT" PARAMETERTYPE="COM.JOANG. ... "
>> Find! (8)<SELECT-</SELECT> 2618 Col
\_ TEXT = " ID="SELECTFILELIST" PARAMETERTYPE="COM.JOANG.FIL ... "
>> Find! (9)<SELECT-</SELECT> 536 Col
\_ TEXT = " ID="SELECTFILEDETAIL" PARAMETERTYPE="INT" RESULT ... "
>> Find! (10)<SELECT-</SELECT> 384 Col
\_ TEXT = " ID="SELECTFILESHAREUSERLISTBYFILE" PARAMETERTYPE= ... "
>> Find! (11)<SELECT-</SELECT> 107 Col
\_ TEXT = " ID="GETTARGETMETAIDX" RESULTTYPE="STRING"> SELEC ... "
```

```
>> Find! (12)<SELECT-</SELECT> 228 Col
\_ TEXT = " ID="CHECKMETA" PARAMETERTYPE="STRING" RESULTTYPE ... "
>> Find! (15)<SELECT-</SELECT> 399 Col
\_ TEXT = " ID="SELECTMETALISTBYFILE" PARAMETERTYPE="INT" RE ... "
>> Find! (16)<SELECT-</SELECT> 1426 Col
\_ TEXT = " ID="SELECTMETAREFLIST" PARAMETERTYPE="COM.JOANG. ... "
>> Find! (17)<SELECT-</SELECT> 305 Col
\_ TEXT = " ID="CHECKMETAMAPPING" PARAMETERTYPE="HASHMAP" RE ... "
>> Find! (18)<SELECT-</SELECT> 241 Col
\_ TEXT = " ID="SELECTMAPPINGLISTBYMETAIDX" PARAMETERTYPE="IN ... "
>> ::::::::: key="<select-</select>" , FIND("TB_FILEMANAGER_META") Total Cnt = 15
>> Find! (1)<INSERT-</INSERT> 453 Col
\_ TEXT = " ID="INSERTMETA" PARAMETERTYPE="HASHMAP"> <![CD ... "
>> Find! (2)<INSERT-</INSERT> 648 Col
\_ TEXT = " ID="INSERTMETAMAPPING" PARAMETERTYPE="HASHMAP"> ... "
>> Find! (3)<INSERT-</INSERT> 375 Col
\_ TEXT = " ID="INSERTMETAEXISTMAPPING" PARAMETERTYPE="HASHMA ... "
>> Find! (4)<INSERT-</INSERT> 332 Col
\_ TEXT = " ID="INSERTSHAREUSER" PARAMETERTYPE="COM.JOANG.FIL ... "
>> Find! (5)<INSERT-</INSERT> 171 Col
\_ TEXT = " ID="UPDATEMERGEMETAMAPPING" PARAMETERTYPE="HASHMA ... "
>> Find! (6)<INSERT-</INSERT> 130 Col
\_ TEXT = " ID="UPDATEMERGEMETA" PARAMETERTYPE="HASHMAP"> UP ... "
>> Find! (7)<INSERT-</INSERT> 104 Col
\_ TEXT = " ID="DELETEMETADETAIL" PARAMETERTYPE="INT"> DELET ... "
>> Find! (8)<INSERT-</INSERT> 659 Col
\_ TEXT = " ID="INSERTFILE" PARAMETERTYPE="HASHMAP"> <![C ... "
>> Find! (9)<INSERT-</INSERT> 635 Col
\_ TEXT = " ID="UPDATEFILE" PARAMETERTYPE="HASHMAP"> <![C ... "
>> Find! (10)<INSERT-</INSERT> 129 Col
\_ TEXT = " ID="UPDATEFILEDETAIL" PARAMETERTYPE="HASHMAP"> U ... "
>> Find! (11)<INSERT-</INSERT> 170 Col
\_ TEXT = " ID="UPDATEFILEMETAMAPPING" PARAMETERTYPE="HASHMAP ... "
>> Find! (12)<INSERT-</INSERT> 270 Col
\_ TEXT = " ID="INSERTMETAREF" PARAMETERTYPE="COM.JOANG.FILE ... "
>> Find! (13)<INSERT-</INSERT> 256 Col
\_ TEXT = " ID="UPDATEMETAREF" PARAMETERTYPE="COM.JOANG.FILE ... "
>> Find! (14)<INSERT-</INSERT> 210 Col
\_ TEXT = " ID="DELETEMETAREF" PARAMETERTYPE="COM.JOANG.FILE ... "
>> ::::::::: key="<insert-</insert>" , FIND("TB_FILEMANAGER_META") Total Cnt = 13
>> Find! (1)<UPDATE-</UPDATE> 130 Col
\_ TEXT = " ID="UPDATEYOUTUBEDOWNLOADYN" PARAMETERTYPE="INT"> ... "
>> Find! (2)<UPDATE-</UPDATE> 239 Col
\_ TEXT = " ID="UPDATEMERGEMETAFROMREF" PARAMETERTYPE="HASHMA ... "
>> Find! (3)<UPDATE-</UPDATE> 233 Col
\_ TEXT = " ID="UPDATEMERGEMETATOREF" PARAMETERTYPE="HASHMAP" ... "
>> ::::::::: key="<update-</update>" , FIND("TB_FILEMANAGER_META") Total Cnt = 3
>> Find! (1)<DELETE-</DELETE> 122 Col
\_ TEXT = " ID="DELETESHAREUSERS" PARAMETERTYPE="INT"> <! ... "
>> Find! (2)<DELETE-</DELETE> 190 Col
\_ TEXT = " ID="DELETEFILEDETAIL" PARAMETERTYPE="INT"> <![CDA ... "
>> Find! (3)<DELETE-</DELETE> 240 Col
\_ TEXT = " ID="DELETEMETAMAPPING" PARAMETERTYPE="HASHMAP"> < ... "
>> Find! (4)<DELETE-</DELETE> 198 Col
\_ TEXT = " ID="DELETEMETAMAPPINGBYFILEIDX" PARAMETERTYPE="IN ... "
>> Find! (5)<DELETE-</DELETE> 199 Col
\_ TEXT = " ID="DALETEMERGEMETAREF" PARAMETERTYPE="HASHMAP"> ... "
```

>> Find! (6)<DELETE-</DELETE> 124 Col
_ TEXT = " ID="DELETEMETAMAPPINGFROMREF" PARAMETERTYPE="INT" ... "
>> Find! (7)<DELETE-</DELETE> 120 Col
_ TEXT = " ID="DELETEMETAMAPPINGTOREF" PARAMETERTYPE="INT"> ... "
>> ::::::::: key="<delete-</delete>" , FIND("TB_FILEMANAGER_META") Total Cnt = 6
:::::::::::: File=metaMapper.xml , Result={<SELECT=15, <DELETE=6, <INSERT=13, <UPDATE=3}::::::::::::

:::::::::::: PROC FILE NAME : test.txt::::::::::::
> fileName Path : C:\HYUNSU\eclipse-showcase-workspace\SocParsing\parsing\test.txt
> Remove Anno Range : test.txt [/* , */]
> Remove Anno Range : test.txt [<!-- , -->]
> Remove Anno Line : test.txt [--]
> Remove Anno Line : test.txt [//]
>> ::::::::: key="<select-</select>" , FIND("TB_FILEMANAGER_META") Total Cnt = 0
>> ::::::::: key="<insert-</insert>" , FIND("TB_FILEMANAGER_META") Total Cnt = 0
>> ::::::::: key="<update-</update>" , FIND("TB_FILEMANAGER_META") Total Cnt = 0
>> ::::::::: key="<delete-</delete>" , FIND("TB_FILEMANAGER_META") Total Cnt = 0
:::::::::::: File=test.txt , Result={<SELECT=0, <DELETE=0, <INSERT=0, <UPDATE=0}::::::::::::

>>>>>>>>>> All File Find ["TB_FILEMANAGER_META"] Result={<SELECT=15, <DELETE=6, <INSERT=13, <UPDATE=3}