006. Mybatis LocalCache

아래의 사용자 데이터 수정 메서드는 두 작업을 한다.

- 사용자 데이터 (이름, 삭제여부, Start - End Date 등)
- 사용자 Role 수정

이 때 각각existProjectUserByPm 메서드를 통해 현재 프로젝트에 남아있는 PM이 있는지 보고 없다면 예외를 던져 모든 트랜잭션을 롤백시킨다.

즉, 하나의 프로젝트 내에서 PM Role을 가진 사용자가 반드시 한 명 이상 있어야 한다.

```
public class ProjectResourceService {

   @Transactional
   public void save() {
      saveUser();
            saveUserRoles();
   }

   public void saveUser() {
     // 사용자 데이터 수정 로직 ...

     if (!existProjectUserByPm(project.getProjectKey())) {
          throw new IllegalStateException(""Last PM cannot be deleted.);
        }
   }

   public void saveUserRoles() {
     // 사용자 Role 수정 로직...

     if (!existProjectUserByPm(project.getProjectKey())) {
          throw new IllegalStateException(""Last PM cannot be deleted.);
        }
   }

      public boolean existProjectUserByPm(int projectKey) {
         return (projectResourceMapper.selectProjectUserByPm(projectKey) > 0);
      }
}

interface PprojectResourceMapper {
   int selectProjectUserByPm(@Param("projectKey") int projectKey);
}
```

이때 만약 프로젝트에 남아있는 사용자의 수를 집계하는 쿼리가 아래와 같이 선언된 경우 Mybatis의 캐시로 인해서 이 로직은 제대로 동작하지 않을 가능성이 크다.

Mybatis PprojectResourceMapper.xml

```
<select id="selectProjectUserByPm" parameterType="int" resultType="int">
   SELECT COUNT(DISTINCT TU.USER_KEY) COUNT
   FROM ~~~~
</select>
```

Mybatis에는 두 가지 내장 캐시가 존재한다.

Local session cache는 무조건 활성화 된다. (on / off 설정이 없음) Second level cacah는 Mapper의 네임스페이스 단위로 동작하고 on / off 할 수 있다.

https://mybatis.org/mybatis-3/sqlmap-xml.html발췌

cache

MyBatis includes a powerful transactional query caching feature which is very configurable and customizable.

By default, just local session caching is enabled that is used solely to cache data for the duration of a session.

실제 코드상에서 확인해보면 아래와 같다.

org.apache.ibatis.executor.BaseExecutor 클래스의 query 메서드를 보면

```
@SuppressWarnings("unchecked")
public <E> List<E> query(MappedStatement ms, Object parameter, RowBounds rowBounds, ResultHandle
  ErrorContext.instance().resource(ms.getResource()).activity("executing a query").object(ms.get
  if (closed) throw new ExecutorException("Executor was closed.");
  if (queryStack == 0 && ms.isFlushCacheRequired()) {
    clearLocalCache();
  }
  List<E> list;
  try {
    queryStack++;
    list = resultHandler == null ? (List<E>) localCache.getObject(key) : null;
    if (list != null) {
      handleLocallyCachedOutputParameters(ms, key, parameter, boundSql);
    } else {
      list = queryFromDatabase(ms, parameter, rowBounds, resultHandler, key, boundSql);
    }
  } finally {
    queryStack--;
  }
  if (queryStack == 0) {
    for (DeferredLoad deferredLoad : deferredLoads) {
      deferredLoad.load();
    }
    deferredLoads.clear(); // issue #601
    if (configuration.getLocalCacheScope() == LocalCacheScope.STATEMENT) {
      clearLocalCache(); // issue #482
    }
  }
  return list;
}
```

ResultHandler 객체가 null이면 localCache에서 먼저 가져오고 값이 없으면 원래의 쿼리를 날리도록 되어있다.

```
try {
  queryStack++;
  list = resultHandler == null ? (List<E>) localCache.getObject(key) : null;
  if (list != null) {
    handleLocallyCachedOutputParameters(ms, key, parameter, boundSql);                    // localCache로 핸들링
  } else {
    list = queryFromDatabase(ms, parameter, rowBounds, resultHandler, key, boundSql);  // 실제 쿼리가 날아가는 부분
  }
} finally {
  queryStack--;
}
```

ResultHandler 객체는 resultMap 혹은 resultType이 객체로 선언된 경우 등에 Mybatis에 의해서 주입되는 객체이다.

즉, resultType이 int, long 등으로 원시타입인 경우 resultHandler는 null이 될 것이며 Local Cache의 대상이 된다.

다시 아래 코드를 보면 PM Role이 할당된 사용자를 삭제하지는 않았으나 PM Role을 가진 마지막 프로젝트 사용자의 Role을 할당 해제하면

캐시 된 결과에 의해 로직이 뚫린다.

```
public class ProjectResourceService {

  @Transactional
  public void save() {
    saveUser();
        saveUserRoles();
  }

  public void saveUser() {
    // 사용자 데이터 수정 로직 ...

    if (!existProjectUserByPm(project.getProjectKey())) {                    // 사용자가 삭제되지 않았으므로 selectProjectUserByPm 결과는 1이며 결과는 캐시된다.
      throw new IllegalStateException(""Last PM cannot be deleted.);
    }
  }

  public void saveUserRoles() {
    // 사용자 Role 수정 로직...                                                  // 사용자 Role이 PM에서 다른 Role로 바뀌었고 이제 프로젝트에 PM Role을 가진 사용자는 없다.

    if (!existProjectUserByPm(project.getProjectKey())) {                    // 쿼리를 실제로 날려보면 결과는 0이 나와서 예외가 터져야 하지만 결과가 캐시되어 1이 나온다.
      throw new IllegalStateException(""Last PM cannot be deleted.);          // 롤백되지 않고 트랜잭션이 정상 커밋 된다.
    }
  }

    public boolean existProjectUserByPm(int projectKey) {
      return (projectResourceMapper.selectProjectUserByPm(projectKey) > 0);
    }
}

interface PprojectResourceMapper {
  int selectProjectUserByPm(@Param("projectKey") int projectKey);
}
```

해결법은 여러 방안이 있을 수 있다.selectProjectUserByPm 쿼리를 집계 쿼리가 아닌 별도의 Dto List로 받도록 해서 List의 size가 0인지 비교하는 것이다.

```
interface PprojectResourceMapper {
   List<UserRole> selectProjectUserByPm(@Param("projectKey") int projectKey);
}

public boolean existProjectUserByPm(int projectKey) {
   return (projectResourceMapper.selectProjectUserByPm(projectKey).size() > 0);
}
```

두 번째는 Mybatis의 select 태그에 선언된 flushCache attribute를 사용한다. (flushCache의 기본 옵션은 false이다.)

```
<!ELEMENT select (#PCDATA | include | trim | where | set | foreach | choose | if | bind)*>
<!ATTLIST select
id CDATA #REQUIRED
parameterMap CDATA #IMPLIED
parameterType CDATA #IMPLIED
resultMap CDATA #IMPLIED
resultType CDATA #IMPLIED
resultSetType (FORWARD_ONLY | SCROLL_INSENSITIVE | SCROLL_SENSITIVE) #IMPLIED
statementType (STATEMENT|PREPARED|CALLABLE) #IMPLIED
fetchSize CDATA #IMPLIED
timeout CDATA #IMPLIED
flushCache (true|false) #IMPLIED
useCache (true|false) #IMPLIED
databaseId CDATA #IMPLIED
lang CDATA #IMPLIED
resultOrdered (true|false) #IMPLIED
resultSets CDATA #IMPLIED
>
```

```
<select id="selectProjectUserByPm" parameterType="int" resultType="int"
        flushCache="true">
    SELECT COUNT(DISTINCT TU.USER_KEY) COUNT
```

flushCache 옵션이 true로 설정된 쿼리는 아래의 쿼리에서 ms.isFlushCacheRequired()가 true로 발생하여 clearLocalCache() 메서드를 통해 캐시가 유지되지 않는다.

따라서 resultHandler가 없더라도 localCache.getObject(key)의 결과가 항상 null 이므로 항상 Database로 질의하게 된다.

```java
public <E> List<E> query(MappedStatement ms, Object parameter, RowBounds rowBounds, Resu
  ErrorContext.instance().resource(ms.getResource()).activity("executing a query").objec
  if (closed) throw new ExecutorException("Executor was closed.");
  if (queryStack == 0 && ms.isFlushCacheRequired()) {
    clearLocalCache();
  }
  List<E> list;
  try {
    queryStack++;
    list = resultHandler == null ? (List<E>) localCache.getObject(key) : null;
    if (list != null) {
      handleLocallyCachedOutputParameters(ms, key, parameter, boundSql);
    } else {
      list = queryFromDatabase(ms, parameter, rowBounds, resultHandler, key, boundSql);
    }
  } finally {
    queryStack--;
  }
```