IUSSE RECORDS

一. 单元测试—SERVICE 层

ISSUE-1：在 CourseService 在测试中发现：红框内 if 条件应为 t<0，老师发布的课程可能在学生还没有选课之前被删除。

CourseServiceImpl.java

```
    /*
    这里要删除course teach 以及所有相关Take
     */
    @Transactional
    public void deleteCourseByTeacher(String username, int course_id) throws Exception {
        UserAndRole user = this.userDao.findUserAndRoleByUserName(username);
        if (user == null) {
            throw new Exception("用户不存在");
        } else {
            int userId = this.teachDao.findUserIdByCourseId(course_id);
            if (userId != user.getUserId()) {
                throw new Exception("没有删除权限");
            } else {
                //int t = this.takesDao.deleteTakesByTeacher(course_id);
                int t = this.takeDao.deleteByCourseId(course_id);
                if (t <= 0) {
                    throw new Exception("数据库异常");
                } else {
                    t = this.teachDao.deleteByCourseId(course_id);
                    if (t <= 0) {
                        throw new Exception("数据库异常");
                    } else {
                        this.courseDao.deleteById(course_id);
                    }
                }
            }
        }
    }
```

二、单元测试—DAO 层

ISSUE-1：查询可用场馆时，查询出的场馆结果少于实际可用场馆。

问题描述：

预约 1 的结束时间与预约 2 的开始时间相同时，两者会产生冲突。

比如，已经存在场馆 1 的预约时间为 09：00—10：00，但是再预约 10：00—11：30 时，查询出的可用场馆中没有 1。

原因：

查询可用场馆 or 场地的时候或者在对比课程和预约纪录的时候，时间比较用的应该是< >而不是<= >=，否则分别从时间 a 结束和时间 a 开始的两个预约会被视为冲突。但是在对比场馆开放时间的时候用的仍是<= >=。原本的 SQL 中全部写成了<= >=，导致查询结果错误。

修改如下：

```
            nativeQuery = true)*/
    //删除别名
    @Query(value = "select gym.gym_id,gym.gym_name,gym.start_time,gym.end_time\n" +
            "from gym\n" +
            "where start_time <= :startTime and end_time >= :endTime " +
            "and  gym_id \n" +
            "not in \n" +
            "(select gym_id \n" +
            "from course\n" +
            "where course.weekday = :weekday and course.end_time > :startTime and course.start_time < :endTime)\n",
            nativeQuery = true)
    List<Gym> queryAvailableGym(String weekday, String startTime, String endTime);
}
```

```
    //删除别名
    @Query(value = "select gym.gym_id,gym.gym_name,gym.start_time,gym.end_time,field.field_id,field.field_name " +
            "from gym natural join field \n" +
            "where gym.start_time <= :startTime and gym.end_time >= :endTime \n" +
            "and (gym_id, field_id) not in \n" +
            "\t( select gym_id ,field_id \n" +
            "\t\tfrom reserve r \n" +
            "\t\twhere r.date = :date and r.end_time > :startTime  and r.start_time < :endTime) \n" +
            "and gym_id not in \n" +
            "( select gym_id \n" +
            "\tfrom course \n" +
            "\twhere course.weekday = :weekday and course.end_time > :startTime  and course.start_time < :endTime)",
    nativeQuery = true)
    List<GymAndField> queryAvailableField(String date, String weekday, String startTime, String endTime);
}
```

ISSUE-2：查询可用场馆时，weekday 参数不合规范，返回全部场馆，而不是返回空结果。

问题描述：

当 weekday 参数不合法时，比如给出”周八“，会返回所有场馆，而不是空。

解决：

这个不算是 DAO 层错误，应该加强规约，确保给 DAO 层合法参数。

ISSUE-3：ReserveDao 中查询全部预约纪录时，同一条纪录会返回两次。

测试出错：

原查询语句以及对应在原生 SQL 在 MYSQL 上的查询效果：

```
@Query(value = "select new com.example.demo.vo.ReserveAndUserAndGymAndField" +
        "(r.reserveId,r.userId,u.userName,r.gymId,g.gymName,r.fieldId,f.fieldName,r.date,r.startTime,r.endTime) " +
        "from Reserve r join User u  " +
        "on u.userId = ?1 and r.userId = u.userId " +
        "join Gym g " +
        "on r.gymId = g.gymId " +
        "join Field f " +
        "on r.fieldId = f.fieldId ") 应该再添加一个gymId相等的条件
List<ReserveAndUserAndGymAndField> queryReserveByUser(Integer uid);
//List<Reserve> queryReserveByUser(int uid);
```

```
mysql>
mysql> select
    ->             r.reserve_id,r.user_id,u.user_name,r.gym_id,g.gym_name,r.field_id,f.field_name,r.date,r.start_time,r.end_time
    ->             from Reserve r  inner join User u
    ->             on u.user_id = 3 and r.user_id = u.user_id
    ->              inner join Gym g
    ->             on r.gym_id = g.gym_id
    ->             join Field f
    ->             on r.field_id = f.field_id;
+------------+---------+-----------+--------+--------------------------+----------+------------+------------+------------+----------+
| reserve_id | user_id | user_name | gym_id | gym_name                 | field_id | field_name | date       | start_time | end_time |
+------------+---------+-----------+--------+--------------------------+----------+------------+------------+------------+----------+
|          1 |       3 | Tom       |      2 | 中北体育馆羽毛球馆       |        1 | 1号场地    | 2019-12-23 | 08:00      | 12:00    |
|          1 |       3 | Tom       |      2 | 中北体育馆羽毛球馆       |        1 | 1号场地    | 2019-12-23 | 08:00      | 12:00    |
+------------+---------+-----------+--------+--------------------------+----------+------------+------------+------------+----------+
2 rows in set (0.03 sec)
```

问题原因：

查询出的gym_id会重复出现在属性中(大约SQL的JOIN语句不去重的问题)。

解决方法：

在连续进行三次 join 时，为最后一次 join 添加一个 g.gym_id = f.gym_id 条件即可。

```
@Query(value = "select new com.example.demo.vo.ReserveAndUserAndGymAndField" +
        "(r.reserveId,r.userId,u.userName,r.gymId,g.gymName,r.fieldId,f.fieldName,r.date,r.startTime,r.endTime) " +
        "from Reserve r join User u  " +
        "on u.userName = ?1 and r.userId = u.userId " +
        "join Gym g " +
        "on r.gymId = g.gymId " +
        "join Field f " +
        "on g.gymId = f.gymId and r.fieldId = f.fieldId ")
List<ReserveAndUserAndGymAndField> queryReserveInfoByUser(String name);
//List<Reserve> findByUserName(String name);


@Query(value = "select new com.example.demo.vo.ReserveAndUserAndGymAndField" +
        "(r.reserveId,r.userId,u.userName,r.gymId,g.gymName,r.fieldId,f.fieldName,r.date,r.startTime,r.endTime) " +
        "from Reserve r join User u  " +
        "on u.userId = ?1 and r.userId = u.userId " +
        "join Gym g " +
        "on r.gymId = g.gymId " +
        "join Field f " +
        "on g.gymId = f.gymId and r.fieldId = f.fieldId ")
List<ReserveAndUserAndGymAndField> queryReserveByUser(Integer uid);
//List<Reserve> queryReserveByUser(int uid);
```

ISSUE-4 ：.TeachDao 中 int findUserIdByCourseId(int courseId 方法调用时无有效响应。

原因：

int findUserIdByCourseId(int courseId)；在没有自定义 HQL 的情况下，由 JPA 直接解析会返回 User 类，而不是 userId，出错。

解决方法：

为其补上@Query()，即自己写上查询语句。

另外：由于该方法在查询不到结果时不会返回 int，而是返回空，为了避免程序出错，为方法添加异常。

```
//int queryUserId(int courseId);
@Query(value = "select user_id from teach where course_id = ?1",nativeQuery = true)
int findUserIdByCourseId(int courseId) throws RuntimeException;
```

三、单元-CONTROLLER 层

ISSUE-1:.只 mock url，无法绕过安全验证。

问题描述：

controler 层的测试，如果只是像以前一样只 mock url，无法跳过安全验证。

解决方法：

需要加上@MockUser，并为用户设定对应的权限，确保其可以或者不可以访问该接口。

```java
@Test
@WithMockUser(username="admin",roles = {"ADMIN"})
void queryAllCourse_admin_ok() throws Exception{
    mockMvc.perform(post( urlTemplate: "/admin/add/announce")
    .param( name: "time", ...values: "2019-12-29")
    .param( name: "content", ...values: "这是一条控制层测试通知"))
            .andExpect(status().isOk());
    verify(announceService, times( wantedNumberOfInvocations: 1)).addAnnounce(any());
}
@Test
@WithMockUser(username="tourist",roles = {"TOURIST"})
void queryAllCourse_tourist_forbidden() throws Exception{
    mockMvc.perform(post( urlTemplate: "/admin/add/announce")
            .param( name: "time", ...values: "2019-12-29")
            .param( name: "content", ...values: "这是一条控制层测试通知"))
            .andExpect(status().isForbidden());
    verify(announceService, times( wantedNumberOfInvocations: 0)).addAnnounce(any());
}
```

四、集成测试

ISSUE-1：在 TeacherController 接口测试中发现：当老师查询可用场馆时，程序出错：无法准备 SQL。

发现问题的地方：集成测试中

问题出现的地方：DAO 层

错误原因：SQL 中使用了别名

解决方法：在 GymDao 中删除 SQL 语句的别名。

```java
//删除别名
@Query(value = "select gym.gym_id,gym.gym_name,gym.start_time,gym.end_time\n" +
        "from gym\n" +
        "where start_time <= :startTime and end_time >= :endTime " +
        "and  gym_id \n" +
        "not in \n" +
        "(select gym_id \n" +
        "from course\n" +
        "where course.weekday = :weekday and course.end_time > :startTime and course.start_time < :endTime)\n",
        nativeQuery = true)
List<Gym> queryAvailableGym(String weekday, String startTime, String endTime);
```

```
//删除别名
@Query(value = "select gym.gym_id,gym.gym_name,gym.start_time,gym.end_time,field.field_id,field.field_name " +
        "from gym natural join field \n" +
        "where gym.start_time <= :startTime and gym.end_time >= :endTime \n" +
        "and (gym_id, field_id) not in \n" +
        "\t( select gym_id ,field_id \n" +
        "\t\tfrom reserve r \n" +
        "\t\twhere r.date = :date and r.end_time > :startTime  and r.start_time < :endTime) \n" +
        "and gym_id not in \n" +
        "( select gym_id \n" +
        "\tfrom course \n" +
        "\twhere course.weekday = :weekday and course.end_time > :startTime  and course.start_time < :endTime)",
 nativeQuery = true)
List<GymAndField> queryAvailableField(String date, String weekday, String startTime, String endTime);
```

这个错误部分报错如下：

"timestamp": "2019-12-30T08:21:01.754+0000",
"status": 500,
"error": "Internal Server Error",
"message": "could not execute query; SQL [select gym.gym_id as gymId,gym.gym_name as gymName,gym.start_time as startTime,gym.end_time as endTime\nfrom gym\nwhere start_time <= ? and end_time >= ? and gym_id \nnot in \n(select gym_id \nfrom course\nwhere course.weekday = ? and course.end_time > ? and course.start_time < ?)\n]; nested exception is org.hibernate.exception.SQLGrammarException: could not execute query",

由于 DAO 层单元测试已经完成，我没有想到是 DAO 层语法的问题。

但是它确实有语法异常，那可能是前端参数形式不对，比如传了空值，SQL 也无法组装。

但是经过测试，发现前端参数正常。

最后我又看到这么一条报错：

java.sql.SQLException: Column 'gym_id' not found.

我重新为对应的实体类构造函数加上了 gym_id 这个参数，仍然报错。

最后最后才发现，是 SQL 别名的问题。

疑问：在单元测试过程中（h2 数据库），该缺陷没有报错。但在进行接口测试时，出现 sql 语句执行错误的报错。

ISSUE-2: 在 Student Controller 接口测试中发现：某个学生尝试加入自己已经加入的课程，仍然会提示加入成功

问题描述：如标题。

其他：该问题并不影响程序正确性（添加前后数据库内容并不发生变化 ）。但是可以进行优化，可以在前端进行限制。

```
mysql> select * from take;
+---------+-----------+
| user_id | course_id |
+---------+-----------+
|       4 |         1 |
|       4 |         2 |
|       5 |         2 |
|       6 |         3 |
+---------+-----------+
4 rows in set (0.01 sec)

mysql> select * from take;
+---------+-----------+
| user_id | course_id |
+---------+-----------+
|       4 |         1 |
|       4 |         2 |
|       5 |         2 |
|       6 |         3 |
+---------+-----------+
4 rows in set (0.00 sec)
```

ISSUE-3: 在 Student Controller 接口测试中发现：某个学生尝试删除自己已经加入的课程，程序会出错。

问题描述：

学生删除已加入的课程 1，程序出错。

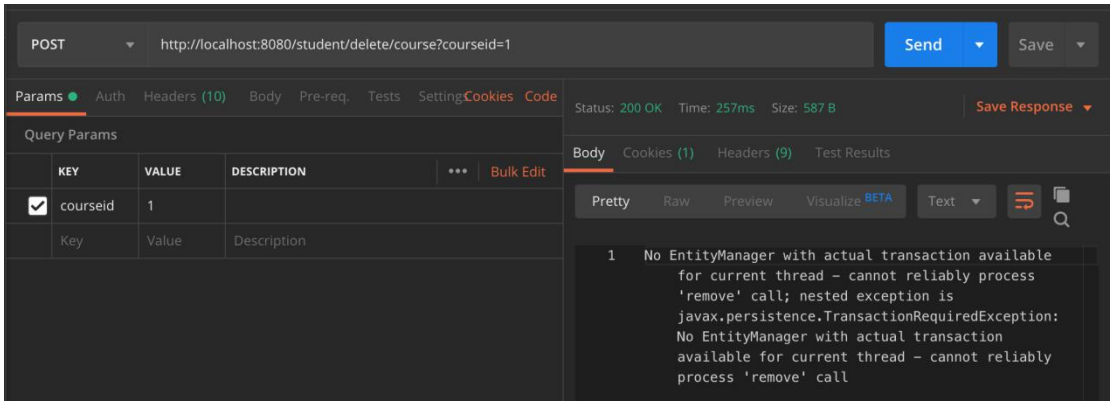
观察了打印出的 sql 语句，并没有执行 delete 语句

原因：

这个异常可能是由于在需要事务的方法上，没有开启事务，结果就操作需要事务的方法比如保存，修改数据库数据方法。

解决方法：

对 service 层的删除方法加上@Transactional 的注解。



ISSUE-4：在 Student Controller 接口测试中发现：某个学生尝试加入不存在课程，程序会返回成功。

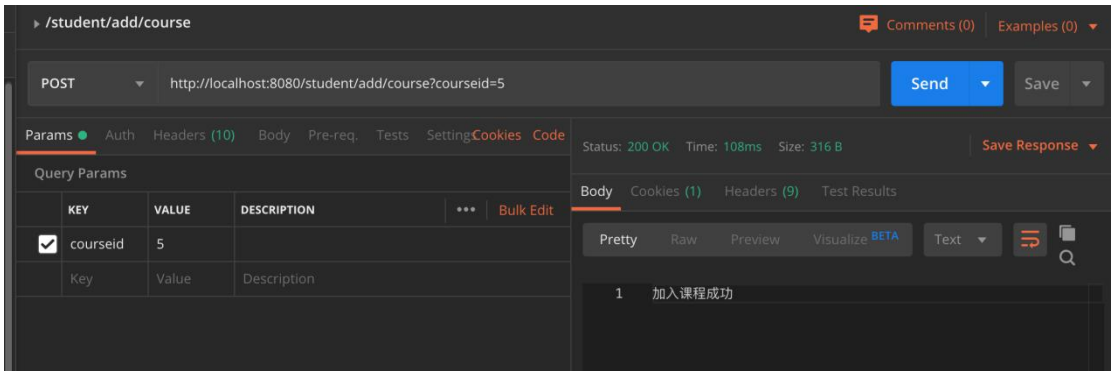
问题描述：如上。

原因：

数据库不会对保存 Take 关系进行限制。

解决方法：

在前端加上限制，只能加入已经存在的课程（即从已经存在的课程中进行挑选）。



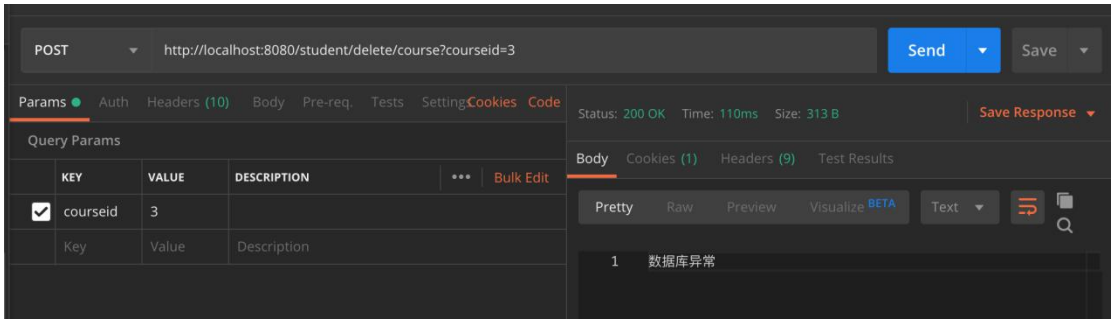
ISSUE-5：在 Student Controller 接口测试发现：某个学生尝试删除自己未加入的课程，返回“数据库异常”。

问题描述：如上。但是实际上并非数据库异常，而是逻辑异常。

原因：

数据库删除结果数量为 0，抛出异常。

解决方法：在前端加上限制，只能删除已经加入的课程。



ISSUE-6：在 Admin Controller 接口测试中发现：1.数据的 ID 不对 2.数据的时间属性格式不对

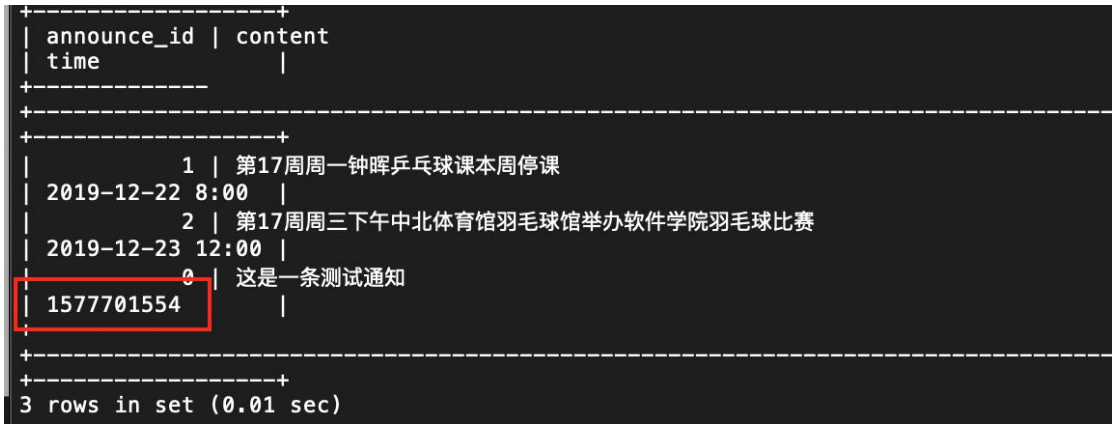
问题 1：数据在 ID 不对

原因：

数据库表 announce 没有自增（constraint.sql 为所有表添加自增，但是 announce 写错了，所以这个表没有添加成功）

解决方法：给 announce 加上自增即可。

问题 2：数据的时间属性格式不对（一个可能的缺陷）



原因：系统没有对传入的时间参数做形式上的限制。

这个限制可以在前端/CONTROLLER/SERVICE 做。如果没有任何一层做，那这里就是一个缺陷。

总结

在单元测试和集成测试中

已经发现并且被解决的问题：

一. ISSUE-1
二. ISSUE-1 ISSUE-3 ISSUE-4
三. ISSUE-1
四. ISSUE-3

已经发现并且需要前端加强约束的问题：

二.
ISSUE-2 weekday 参数需要确保为“周一”到“周日”，且格式正确。
四.

ISSUE-2 在前端进行限制，将学生已经加入的课程从可选课程列表中删除，避免重复添加。（不加也可，不影响程序正确性）

ISSUE-4 在前端进行限制，要求学生只能从已有的课程中选择课程并加入。避免学生直接输入课程 ID（课程 ID 可能不存在）要求加入。

ISSUE-5 同 5.在前端进行限制，要求学生只能从自己的课程中选择课程并删除。避免学生直接输入课程 ID 要求删除对该课程的 take 关系。
（不会对数据库造成影响，但是 dao 层会抛出异常。）

ISSUE-6 需要在前端对 date/starttime/endtime 等参数的格式进行限制。

Date：“YYYY-MM-DD HH:MM”

Starttime：”HH:MM”

Endtime：”HH:MM”