DBS hw05

3200102888

5.6

1.

```sql
create trigger trigger_1 after insert on depositor
referencing new row as nrow
for each row
insert into branch_cust
    select branch_name, nrow.customer_name
    from account
    where account_number = nrow.account_number

```

2.

```sql
create trigger trigger_1 after insert on account
referencing new row as nrow
for each row
insert into branch_cust
    select nrow.branch_name, customer_name
    from depositor
    where customer_name = nrow.customer_name
```

5.15

1.

```sql
create function avg_salary (company_name varchar(20))
returns integer
begin
declare avg_sal integer;
select avg(salary) into d_count
from works
where works.company_name = company_name
group by company_name
return avg_sal
end
```

2.

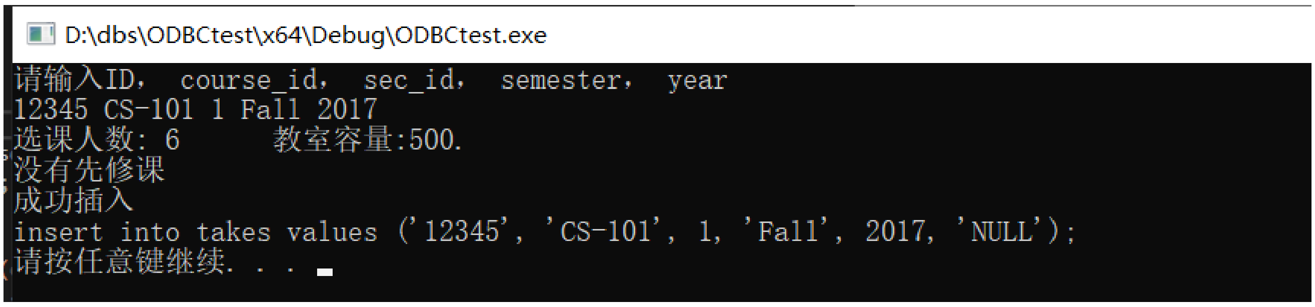
```sql
select company_name from works
where avg_salary(works.company_name) > avg_salary("First Bank")
```

exercise

写一个嵌入SQL/ODBC程序或Stored Procedure，保存一位同学的一门选课信息，需检查不能有冲突的上课时间；所有先修课必须通过；教室容量必须够。如果以上条件不满足则失败。

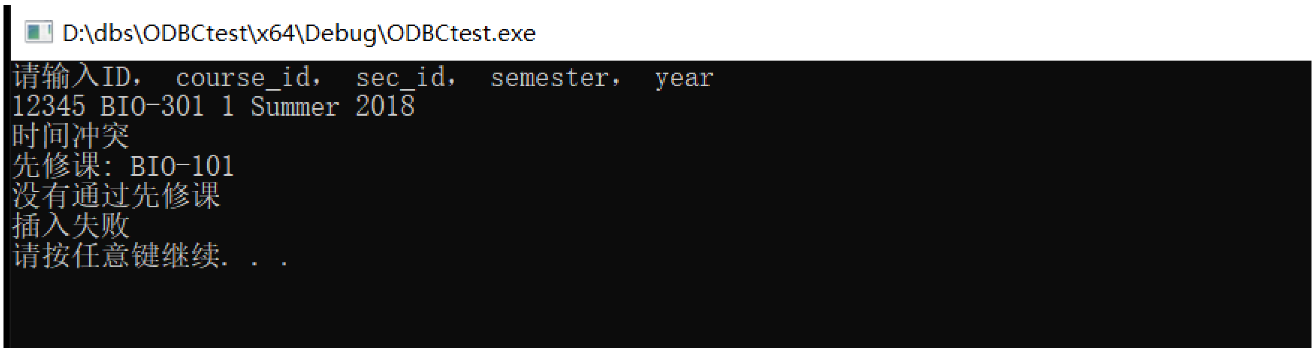
```cpp
#include <cstdio>
#include "mysql.h"
#include <iostream>
#include <string>
using namespace std;
int main()
{
    MYSQL mysql;    //一个数据库结构体
    MYSQL_RES* res; //一个结果集结构体
    MYSQL_ROW row;  //char** 二维数组，存放记录
    int res1;
    mysql_init(&mysql);
    mysql_options(&mysql, MYSQL_SET_CHARSET_NAME, "gbk");
    string ID,course_id,semester;
    int year,sec_id;
    string grade;
    bool flag = true;
    cout << "请输入ID， course_id， sec_id， semester， year" << endl;
    cin >> ID >> course_id >> sec_id >> semester >> year;
    grade = "NULL";
    if (mysql_real_connect(&mysql, "localhost", "root", "111111",
"university", 3306, NULL, CLIENT_MULTI_RESULTS) == NULL)
        printf("连接失败！ \\n");
    char s[500];
    //教室容量足够
    sprintf_s(s, "select count(distinct ID),room_number,building,
capacity,course_id,semester,year  from takes natural join section
natural join classroom where course_id='%s' and sec_id=%d and
semester='%s' and year=%d  group by room_number,
capacity,course_id,semester,year,building;", course_id.c_str(),
sec_id, semester.c_str(), year);
    res1 = mysql_query(&mysql, s);
    res = mysql_store_result(&mysql);
    if (res1) {
        fprintf(stderr, "error %d: %s\n", mysql_errno(&mysql),
mysql_error(&mysql));
    }
    while (row = mysql_fetch_row(res)) {
        printf("选课人数: %d\t 教室容量:%d. \n",
atoi(row[0]),atoi(row[3]));
        if (atoi(row[0]) >= atoi(row[3])) {
            cout << "教室已满! " << endl;
            flag = false;
        }
    }
    //没有时间冲突
    sprintf_s(s, "select * from takes natural join section where semester='%s' and year=%d and sec_id=%d and time_slot_id=(select time_slot_id from section where course_id='%s');"
        , semester.c_str(),year, sec_id, course_id.c_str());
    mysql_free_result(res);
    res1 = mysql_query(&mysql, s);
    res = mysql_store_result(&mysql);
    if (res1) {
        fprintf(stderr, "error %d: %s\n", mysql_errno(&mysql), mysql_error(&mysql));
    }
    if (res->row_count == 0) {
        cout << "时间冲突" << endl;
        flag = false;
    }
    mysql_free_result(res);
    //先修课通过
    char tmps[500];
    sprintf_s(tmps, "select prereq_id from prereq where course_id = '%s';", course_id.c_str());
    res1 = mysql_query(&mysql, tmps);
    res = mysql_store_result(&mysql);
    if (res1) {
        fprintf(stderr, "error %d: %s\n", mysql_errno(&mysql), mysql_error(&mysql));
    }
    if (res->row_count) {
        row = mysql_fetch_row(res);
        cout << "先修课: " << row[0] << endl;
        mysql_free_result(res);
        sprintf_s(s, "select * from takes where (select prereq_id from prereq where course_id = '%s') in (select course_id from takes where ID='%s' and grade <>'F');", course_id.c_str(), ID.c_str());
        res1 = mysql_query(&mysql, s);
        res = mysql_store_result(&mysql);
        if (res->row_count == 0) {
            cout << "没有通过先修课" << endl;
            flag = false;
        }
    }
    else {
        cout << "没有先修课" << endl;
    }
    if (flag){
            cout << "成功插入" << endl;
            sprintf_s(s, "insert into takes values ('%s', '%s', %d, '%s', %d, '%s');",ID.c_str(), course_id.c_str(), sec_id, semester.c_str(), year, grade.c_str());
            cout << s << endl;
            res1 = mysql_query(&mysql, s);
            if (res1) {
                fprintf(stderr, "error %d: %s\n", mysql_errno(&mysql), mysql_error(&mysql));
            }
        }
    else
            cout << "插入失败" << endl;
    mysql_free_result(res);
    mysql_close(&mysql);
    system("pause");
    return 0;
}

```

测试：使用sql.js demo: Online SQL interpreter (db-book.com)中的university数据库数据。



D:\dbs\ODBCtest\x64\Debug\ODBCtest.exe

```
请输入ID, course_id, sec_id, semester, year
12345 CS-101 1 Fall 2017
选课人数: 6      教室容量:500.
没有先修课
成功插入
insert into takes values ('12345', 'CS-101', 1, 'Fall', 2017, 'NULL');
请按任意键继续. . .
```



D:\dbs\ODBCtest\x64\Debug\ODBCtest.exe

```
请输入ID, course_id, sec_id, semester, year
12345 BIO-301 1 Summer 2018
时间冲突
先修课: BIO-101
没有通过先修课
插入失败
请按任意键继续. . .
```