|  |  |  |
| --- | --- | --- |
|  |  | |
| 数据库系统课程设计报告 | | |
| 小组名称：小组成员：张三（组长）、李四撰写时间：2021年1月10日文档版本：V1.0 | |  |

目录页

一、成员分工

成员1：张三（学号）

职位：组长/组员

任务分工：

任务分工撰写要求：

* 详细描述个人任务，与其他人协作情况。
* 要分条清晰列举。
* 反面例子：负责\*\*\*业务的设计工作-设计内容含糊，没有标明具体哪个业务的哪块设计工作，也没有提及和其他人协作情况；负责ppt的设计和测试工作-没有完成数据库课程设计的培养目标，组内每个人都要有有效数据库设计任务。
* 小四，宋体，1.5倍行距。

|  |  |  |  |
| --- | --- | --- | --- |
| 任务  分工 | 马欣萌 | 宫珍妮 | 王昕颖 |
| 数据字典抽取工作 | √ | √ | √ |
| 研究生助教工作的ER图 | √ |  |  |
| 研究生学术交流情况的ER图 |  | √ |  |
| 研究生参与项目情况的ER图 |  |  | √ |
| 全局ER图整合 | √ |  |  |
| 全局ER图分析和修改 | √ | √ | √ |
| 业务的数据库逻辑结构设计 | √ |  |  |
| 设计测试数据 |  |  | √ |
| 测试数据填充 | √ |  |  |
| 已给出语句的SQL语句编写操作 | √ | √ | √ |
| 自行设计的SQL语句的编写操作 |  | √ |  |
| 数据库持久层的设计和实现 | √ |  |  |
| 数据库持久层的测试业务代码 | √ |  |  |
| 每名导师项目经费总剩余情况的视图 |  |  | √ |
| 每名教师已经确认的参与学术交流学生数量的视图 |  | √ |  |
| 每名教师学生参加助教课程情况的视图 | √ |  |  |
| 触发器编写 | √ | √ | √ |
| 过程材料的总结和成果答辩准备 | √ |  |  |

二、需求分析

撰写要求：

* 描述每个业务的主要功能。
* 以业务划分子系统，明确子系统的功能。
* 根据功能分析和列举涉及的数据项。
* 数据项的格式为：数据项名称，数据项粗略类型（字符型、数字型、日期时间型，其他型）、数据项的简要描述。
* 描述与数据相关的非功能性分析，如安全性分析、完整性约束等。
* 小四，宋体，1.5倍行距。

1.实体参与项目认定表（参与项目id，承担工作，开始时间，结束时间，审批状态，审批时间，折合费用，是否为负责人）

2.实体学术交流活动表（学术交流活动id，学术交流活动名称，活动地点，报告英文名称，报告中文名称，国家，省，市，学术交流质量，参会照片，备注，审核状态，导师审批时间，学科负责人审批时间）

3.学科（学科id，学科名，学科类别名称，学科简介）

4.学科负责人（负责人id，是否为学科负责人）

5.导师（导师id，是否为导师）

6.志愿选择（志愿id，志愿选择时间）

7.授课教师（授课教师id，是否为授课教师）

8.教师（教师id，教师姓名，职称，电话，出生日期，邮箱地址，性别）

9.用户（用户id，用户密码，上次登录时间，上次登录地点）

10.用户角色（用户角色id，用户角色名，对应角色id）

11.研究生（研究生学号，研究生姓名，联系方式，出生日期，性别，邮箱）

12.研究生助教评定（研究生助教评定id，助教工作自述，授课教师评价，授课教师评价结果，审核结果，审核时间）

13.研究生培养管理员（管理员id，管理员姓名，职务，联系方式，邮箱）

14.课程（课程id，课程名，课程性质，课程开始时间，课程结束时间，授课对象，选课人数，学时）

15.项目（项目id，项目名称，开始时间，结束时间，经费数量，审批状态，审批时间）

16.项目类别（项目类别id，项目类别名称）

三、数据库概念结构设计

撰写要求：

* 给出每个业务或者每个子系统的局部E-R图。
* 给出整个系统的全局E-R图，注意E-R图排版要清晰，字号适中，保证可以看清，全局E-R图页面可以使用横版排版。
* 小四，宋体，1.5倍行距。

1. 每个业务的局部ER图

研究生助教工作

ER图part1

研究生学术交流情况

ER图paart2

研究生参与项目情况

ER图part3

1. 整个系统的ER图



四、数据库逻辑结构设计

撰写要求：

* 给出关系关系模式，标注关系模式的主码和外码。
* 分析各个关系模式的范式级别。
* 小四，宋体，1.5倍行距。

|  |  |  |  |
| --- | --- | --- | --- |
| 关系模式 | 主码 | 外码 | 范式级别 |
| 表助教选择（课程id，研究生助教评定id） | 课程id，研究生助教评定id | 课程id，研究生助教评定id | 3NF |
| 表参与项目认定表（参与项目id，导师id，项目id，承担工作，开始时间，结束时间，审批状态，审批时间，折合费用，是否为负责人） | 参与项目id | 导师id，项目id | 3NF |
| 表学术交流活动表（学术交流活动id，负责人id，研究生学号，管理员id，导师id，学术交流活动名称，报告英文名称，报告中文名称，国家，省，市，学术交流质量，参会照片，备注，审核状态，导师审批时间，学科负责人审批时间） | 学术交流活动id | 负责人id，研究生学号，管理员id，导师id | 3NF |
| 表学科（学科id，负责人id，学科名，学科类别名称，学科简介） | 学科id | 负责人id | 3NF |
| 表学科负责人（负责人id，教师id，学科id，管理员id，是否为学科负责人） | 负责人id | 教师id，学科id，管理员id | 3NF |
| 表导师（导师id，教师id，管理员id，是否为导师） | 导师id | 教师id，管理员id | 3NF |
| 表志愿选择（志愿id，课程id，研究生学号，志愿选择时间） | 志愿id | 课程id，研究生学号 | 3NF |
| 表授课教师（授课教师id，管理员id，教师id，是否为授课教师） | 授课教师id | 管理员id，教师id | 3NF |
| 表授课教师教授课程（授课教师id，课程id） | 授课教师id，课程id | 授课教师id，课程id | 3NF |
| 表教师（教师id，教师姓名，职称，电话，出生日期，邮箱地址，性别） | 教师id | 无 | 3NF |
| 表教师与学科（学科id，教师id） | 学科id，教师id | 学科id，教师id | 3NF |
| 表用户（用户id，用户角色id，用户密码，上次登录时间，上次登录地点） | 用户id | 用户角色id | 3NF |
| 表用户角色（用户角色id，用户角色名，对应角色id） | 用户角色id | 无 | 3NF |
| 表研究生（研究生学号，管理员id，教师id，研究生姓名，联系方式，出生日期，性别，邮箱） | 研究生学号 | 管理员id，教师id | 3NF |
| 表研究生助教评定（研究生助教评定id，管理员id，研究生学号，授课教师id，助教工作自述，授课教师评价，授课教师评价结果，审核结果，审核时间） | 研究生助教评定id | 管理员id，研究生学号，授课教师id | 3NF |
| 表研究生参与项目（研究生学号，参与项目id） | 研究生学号，参与项目id | 研究生学号，参与项目id | 3NF |
| 表研究生培养管理员（管理员id，管理员姓名，职务，联系方式，邮箱） | 管理员id | 无 | 3NF |
| 表课程（课程id，课程名，课程性质，课程开始时间，课程结束时间，授课对象，选课人数，学时） | 课程id | 无 | 3NF |
| 表项目（项目id，项目类别id，管理员id，导师id，项目名称，开始时间，结束时间，经费数量，审批状态，审批时间） | 项目id | 项目类别id，管理员id，导师id | 3NF |
| 表项目类别（项目类别id，项目类别名称） | 项目类别id | 无 | 3NF |

关系模式的图片化表示

五、数据库实施

撰写要求：

* 提供使用的数据库管理系统基本信息，包括软件名称、版本号和其他软硬件环境信息等。
* 使用表格列举每个关系模式的字段名称、字段类型、字段精度、字段约束（主码、外码、自定义约束）。
* 围绕数据安全、数据备份和其他等角度，描述系统可采取的安全策略和风险控制策略。反例：数据安全照搬教材或网上的内容，没有与业务系统结合；撰写内容含糊，仅用1-2句话糊弄了事，可操作性较差。
* 围绕任务书要求，给出每个视图的任务要求和视图编写的SQL语句。
* 围绕任务书要求，给出编写触发器的SQL语句。
* 小四，宋体，1.5倍行距。

使用的软件：

1. 名称：Microsoft SQL Server 2018
2. 版本：2018版
3. 数据库名称：database\_english
4. 用户名：localhost
5. 密码：123456

参与项目认定：



导师：

教师：



教师与学科：



课程：



授课教师：



授课教师教授课程



项目：



项目类别



学科：



学科负责人：



学术交流活动：



研究生：



研究生参与项目：



研究生培养管理员：



研究生助教评定：



用户：

用户角色：



志愿选择：



助教选择：



视图：

每名教师学生参加助教课程情况的视图

/\*每名教师学生参加助教课程情况（教师名称，助教学生数量，助教课程）。\*/

SELECT dbo.teacher.teacher\_name,dbo.course.course\_name, COUNT(dbo.assistant\_chose.assessment\_id) AS num

FROM dbo.teacher INNER JOIN

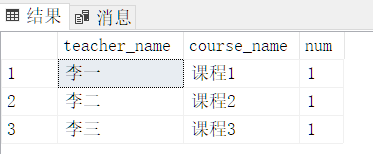
dbo.instructor ON dbo.teacher.teacher\_id = dbo.instructor.teacher\_id INNER JOIN

dbo.teach ON dbo.instructor.instructor\_id = dbo.teach.instructor\_id INNER JOIN

dbo.course ON dbo.teach.course\_id = dbo.course.course\_id INNER JOIN

dbo.assistant\_chose ON dbo.course.course\_id = dbo.assistant\_chose.course\_id

GROUP BY dbo.teacher.teacher\_name, dbo.course.course\_name



每名导师项目经费总剩余情况

/\*每名导师项目经费总剩余情况（教师名称，项目经费剩余情况）\*/

create view rel(tname,rel) AS

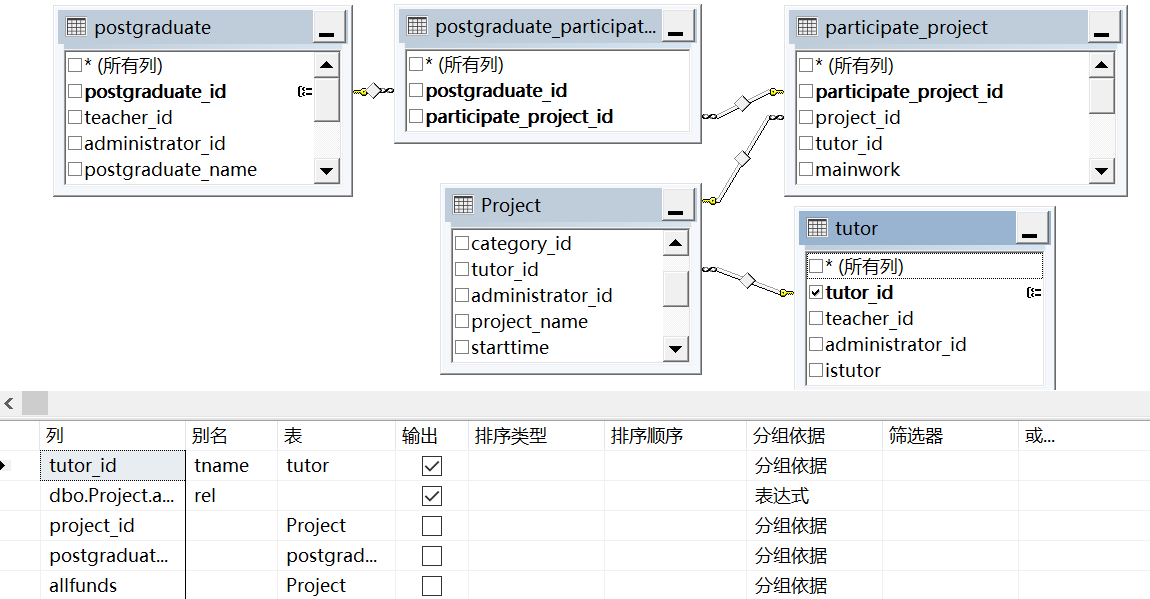
select tutor.tutor\_id,Project.allfunds-sum(costequivalent) as rel

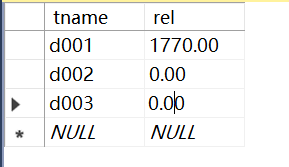
from postgraduate,participate\_project,postgraduate\_participate\_project,Project,tutor

where postgraduate.postgraduate\_id=postgraduate\_participate\_project.postgraduate\_id and postgraduate\_participate\_project.participate\_project\_id=participate\_project.participate\_project\_id

and participate\_project.project\_id=Project.project\_id and tutor.tutor\_id=Project.tutor\_id

group by tutor.tutor\_id,Project.project\_id,postgraduate.postgraduate\_id,Project.allfunds





六、数据库测试

撰写要求：

* 提供每个关系模式的DDL语句。
* 填充测试数据后，根据任务书中SQL语句撰写任务要求，给出每个SQL语句的编写需求和相关SQL语句。
* 总结各SQL语句反馈结果是否满足预期测试要求。
* 小四，宋体，1.5倍行距。

DDL语句

create table Project

(

project\_id char(10) not null

constraint PK\_PROJECT

primary key nonclustered,

category\_id char(10)

constraint FK\_PROJECT\_PROJECT\_C\_CATEGORY

references category,

tutor\_id char(10)

constraint FK\_PROJECT\_IDENTIFY\_\_TUTOR

references tutor,

administrator\_id char(10)

constraint FK\_PROJECT\_AD\_APPROV\_ADMINIST

references administrators,

project\_name varchar(256) not null,

starttime date not null,

endtime date not null,

allfunds decimal(8, 2) not null,

approve\_status int not null,

approve\_time datetime

)

go

create index project\_category\_FK

on Project (category\_id)

go

create index identify\_itemsandfunds\_FK

on Project (tutor\_id)

go

create index ad\_approve\_project\_FK

on Project (administrator\_id)

go

create table academic\_exchange

(

academic\_exchange\_id char(10) not null

constraint PK\_ACADEMIC\_EXCHANGE

primary key nonclustered,

administrator\_id char(10)

constraint FK\_ACADEMIC\_MANAGE\_AC\_ADMINIST

references administrators,

postgraduate\_id char(10)

constraint FK\_ACADEMIC\_POSTGRADU\_POSTGRAD

references postgraduate,

tutor\_id char(10)

constraint FK\_ACADEMIC\_TUTOR\_AUD\_TUTOR

references tutor,

director\_id char(10)

constraint FK\_ACADEMIC\_DIRECTOR\_\_DIRECTOR

references director,

academic\_exchange\_name varchar(256) not null,

report\_englishname varchar(256) not null,

report\_chinesename varchar(256) not null,

country varchar(256),

province varchar(256),

city varchar(256),

quality int,

activity\_photo varchar(256),

others text,

audit\_status int,

tutor\_audit\_time datetime,

director\_audit\_time datetime

)

go

create index postgraduate\_load\_academic\_FK

on academic\_exchange (postgraduate\_id)

go

create index manage\_academicexchange\_FK

on academic\_exchange (administrator\_id)

go

create index tutor\_audit\_academicexchange\_FK

on academic\_exchange (tutor\_id)

go

create index director\_academic\_FK

on academic\_exchange (director\_id)

go

create table administrators

(

administrator\_id char(10) not null

constraint PK\_ADMINISTRATORS

primary key nonclustered,

administrator\_name varchar(256) not null,

administrator\_work varchar(256) not null,

administrator\_phone varchar(50),

administrator\_email varchar(256)

)

go

create table assistant\_assessment

(

assessment\_id char(10) not null

constraint PK\_ASSISTANT\_ASSESSMENT

primary key nonclustered,

postgraduate\_id char(10)

constraint FK\_ASSISTAN\_FINAL\_CHO\_POSTGRAD

references postgraduate,

administrator\_id char(10)

constraint FK\_ASSISTAN\_AD\_APPROV\_ADMINIST

references administrators,

instructor\_id char(10)

constraint FK\_ASSISTAN\_LOAD\_EXAM\_INSTRUCT

references instructor,

self\_account text,

instructor\_evaluate text,

evaluate\_result int not null,

examine\_result int,

examine\_time datetime

)

go

create index final\_choose\_FK

on assistant\_assessment (postgraduate\_id)

go

create index load\_examine\_FK

on assistant\_assessment (instructor\_id)

go

create index ad\_approve\_examine\_FK

on assistant\_assessment (administrator\_id)

go

create table assistant\_chose

(

course\_id char(10) not null

constraint FK\_ASSISTAN\_ASSISTANT\_COURSE

references course,

assessment\_id char(10) not null

constraint FK\_ASSISTAN\_ASSISTANT\_ASSISTAN

references assistant\_assessment,

constraint PK\_ASSISTANT\_CHOSE

primary key (course\_id, assessment\_id)

)

go

create index assistant\_chose\_FK

on assistant\_chose (course\_id)

go

create index assistant\_chose2\_FK

on assistant\_chose (assessment\_id)

go

create table category

(

category\_id char(10) not null

constraint PK\_CATEGORY

primary key nonclustered,

category\_name varchar(256) not null

)

go

create table course

(

course\_id char(10) not null

constraint PK\_COURSE

primary key nonclustered,

course\_name varchar(256) not null,

course\_nature varchar(256) not null,

course\_starttime date,

course\_endtime date,

course\_audience varchar(256),

course\_studentnum int,

course\_teachtime int

)

go

create table director

(

director\_id char(10) not null

constraint PK\_DIRECTOR

primary key nonclustered,

teacher\_id char(10)

constraint FK\_DIRECTOR\_TEACHER\_D\_TEACHER

references teacher,

administrator\_id char(10)

constraint FK\_DIRECTOR\_BUILDDIRE\_ADMINIST

references administrators,

subject\_id char(10)

constraint FK\_DIRECTOR\_MANAGE\_SUBJECT

references subject,

is\_director int not null

)

go

create index manage\_FK

on director (subject\_id)

go

create index builddirector\_FK

on director (administrator\_id)

go

create index teacher\_director\_FK

on director (teacher\_id)

go

create table instructor

(

instructor\_id char(10) not null

constraint PK\_INSTRUCTOR

primary key nonclustered,

teacher\_id char(10)

constraint FK\_INSTRUCT\_TEACHER\_I\_TEACHER

references teacher,

administrator\_id char(10)

constraint FK\_INSTRUCT\_BUILDINST\_ADMINIST

references administrators,

isinstructor int not null

)

go

create index buildinstructor\_FK

on instructor (administrator\_id)

go

create index teacher\_instructor\_FK

on instructor (teacher\_id)

go

create table participate\_project

(

participate\_project\_id char(10) not null

constraint PK\_PARTICIPATE\_PROJECT

primary key nonclustered,

project\_id char(10)

constraint FK\_PARTICIP\_PROJECT\_P\_PROJECT

references Project,

tutor\_id char(10)

constraint FK\_PARTICIP\_TUTOR\_APP\_TUTOR

references tutor,

mainwork text,

participate\_starttime date,

participate\_endtime date,

participate\_approvestatus int,

participate\_approvetime datetime,

costequivalent decimal(8, 2),

isdirector int

)

go

create index project\_participate\_status\_FK

on participate\_project (project\_id)

go

create index tutor\_approve\_participate\_FK

on participate\_project (tutor\_id)

go

create table postgraduate

(

postgraduate\_id char(10) not null

constraint PK\_POSTGRADUATE

primary key nonclustered,

teacher\_id char(10)

constraint FK\_POSTGRAD\_GUIDANCE\_TEACHER

references teacher,

administrator\_id char(10)

constraint FK\_POSTGRAD\_BUILDSTUD\_ADMINIST

references administrators,

postgraduate\_name varchar(256) not null,

postgraduate\_phone varchar(50),

postgraduate\_email varchar(256),

postgraduate\_birth datetime,

postgraduate\_sex varchar(256)

)

go

create index buildstudent\_FK

on postgraduate (administrator\_id)

go

create index guidance\_FK

on postgraduate (teacher\_id)

go

create table postgraduate\_participate\_project

(

postgraduate\_id char(10) not null

constraint FK\_POSTGRAD\_POSTGRADU\_POSTGRAD

references postgraduate,

participate\_project\_id char(10) not null

constraint FK\_POSTGRAD\_POSTGRADU\_PARTICIP

references participate\_project,

constraint PK\_POSTGRADUATE\_PARTICIPATE\_PR

primary key (postgraduate\_id, participate\_project\_id)

)

go

create index postgraduate\_participate\_project\_FK

on postgraduate\_participate\_project (postgraduate\_id)

go

create index postgraduate\_participate\_project2\_FK

on postgraduate\_participate\_project (participate\_project\_id)

go

create table selection

(

selection\_id char(10) not null

constraint PK\_SELECTION

primary key nonclustered,

postgraduate\_id char(10)

constraint FK\_SELECTIO\_FIRST\_CHO\_POSTGRAD

references postgraduate,

course\_id char(10)

constraint FK\_SELECTIO\_CHOOSE\_COURSE

references course,

selection\_time datetime not null

)

go

create index first\_choose\_FK

on selection (postgraduate\_id)

go

create index choose\_FK

on selection (course\_id)

go

create table subject

(

subject\_id char(10) not null

constraint PK\_SUBJECT

primary key nonclustered,

director\_id char(10)

constraint FK\_SUBJECT\_MANAGE2\_DIRECTOR

references director,

subject\_name varchar(256) not null,

subject\_categoryname varchar(256),

subject\_introduction text

)

go

create index manage2\_FK

on subject (director\_id)

go

create table teach

(

instructor\_id char(10) not null

constraint FK\_TEACH\_TEACH\_INSTRUCT

references instructor,

course\_id char(10) not null

constraint FK\_TEACH\_TEACH2\_COURSE

references course,

constraint PK\_TEACH

primary key (instructor\_id, course\_id)

)

go

create index teach\_FK

on teach (instructor\_id)

go

create index teach2\_FK

on teach (course\_id)

go

create table teacher

(

teacher\_id char(10) not null

constraint PK\_TEACHER

primary key nonclustered,

teacher\_name varchar(256),

teacher\_status varchar(256),

teacher\_phone varchar(256),

teacher\_birth date,

teacher\_email varchar(256),

teacher\_sex varchar(256)

)

go

create table teacher\_subject

(

subject\_id char(10) not null

constraint FK\_TEACHER\_\_TEACHER\_S\_SUBJECT

references subject,

teacher\_id char(10) not null

constraint FK\_TEACHER\_\_TEACHER\_S\_TEACHER

references teacher,

constraint PK\_TEACHER\_SUBJECT

primary key (subject\_id, teacher\_id)

)

go

create index teacher\_subject\_FK

on teacher\_subject (subject\_id)

go

create index teacher\_subject2\_FK

on teacher\_subject (teacher\_id)

go

create table tutor

(

tutor\_id char(10) not null

constraint PK\_TUTOR

primary key nonclustered,

teacher\_id char(10)

constraint FK\_TUTOR\_TEACHER\_T\_TEACHER

references teacher,

administrator\_id char(10)

constraint FK\_TUTOR\_BUILDTUTO\_ADMINIST

references administrators,

istutor int not null

)

go

create index buildtutor\_FK

on tutor (administrator\_id)

go

create index teacher\_tutor\_FK

on tutor (teacher\_id)

go

create table [user]

(

userid char(10) not null

constraint PK\_USER

primary key nonclustered,

usercategory\_id char(10)

constraint FK\_USER\_USER\_CATE\_USERCATE

references usercategory,

password varchar(1024) not null,

lastlogintime datetime,

lastloginip varchar(1024)

)

go

create index user\_category\_FK

on [user] (usercategory\_id)

go

create table usercategory

(

usercategory\_id char(10) not null

constraint PK\_USERCATEGORY

primary key nonclustered,

usercategory\_name varchar(256) not null,

now\_id int not null

)

Go

SQL语句：

/\*登录\*/

SELECT \* FROM [user] WHERE userid='t000000001' and password=123456;

/\*指定学科和日期范围内助教结果为合格的学生信息相关课程信息\*/

SELECT \* FROM dbo.course

where course\_starttime>='2020-02-01' and '2020-07-01'>=course\_endtime and

course\_id in(select course\_id from dbo.assistant\_chose

where assessment\_id in(

select assessment\_id from dbo.assistant\_assessment

where postgraduate\_id in(

select postgraduate\_id from dbo.postgraduate

where teacher\_id in(

select teacher\_id from dbo.teacher\_subject

where subject\_id='s000000001'

)

)

)

)

/\*每名授课教师已经确定的助教总人数\*/

SELECT count (\*) as assistantnum

FROM dbo.assistant\_chose t

WHERE course\_id in (

select course\_id from dbo.teach where instructor\_id in(

select instructor\_id from dbo.instructor where teacher\_id='t000000001'

)

)

/\*指定日期范围内和学科下研究生助教数量\*/

SELECT

count (\*)as post\_assistantnum

FROM dbo.assistant\_chose t

WHERE course\_id in (

select course\_id from dbo.course

where course\_starttime>='2020-02-01' and '2020-07-01'>=course\_endtime and course\_id in(

select course\_id from dbo.teach where instructor\_id in(

select instructor\_id from dbo.instructor where teacher\_id in(

select teacher\_id from dbo.teacher\_subject

where subject\_id='s000000001'

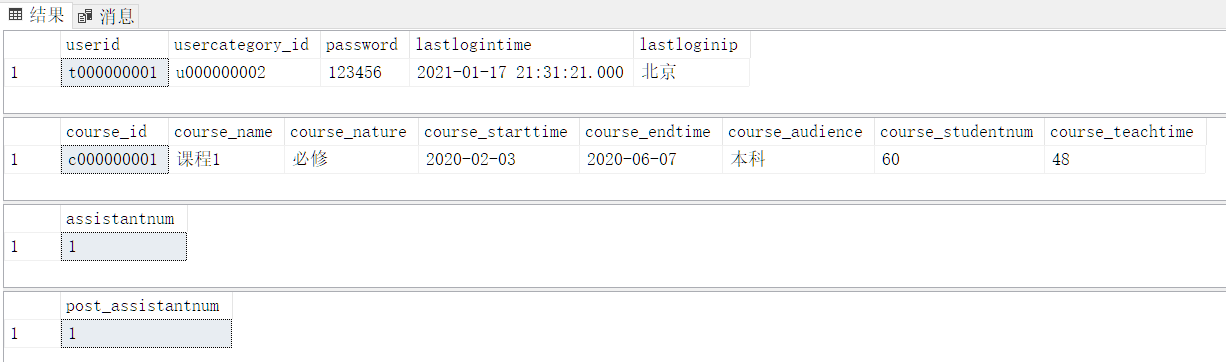
)

)

)

)

结果：满足要求



/\*某个项目已经分配的折合费用\*/

select costequivalent

from participate\_project,Project

where participate\_project.project\_id=Project.project\_id and Project.project\_name='项目2'

/\*指定时间范围内某个学科参与经费的总数量\*/

select allfunds

from Project

where starttime>='2019.9.13' and endtime<='2020.9.13' and

tutor\_id in(select tutor\_id from tutor

where teacher\_id in(select teacher\_id from teacher\_subject where subject\_id='s01'))

/\*每名导师项目经费总剩余情况\*/

select tutor.tutor\_id,Project.allfunds-sum(costequivalent) as rel

from postgraduate,participate\_project,postgraduate\_participate\_project,Project,tutor

where postgraduate.postgraduate\_id=postgraduate\_participate\_project.postgraduate\_id

and postgraduate\_participate\_project.participate\_project\_id=participate\_project.participate\_project\_id

and participate\_project.project\_id=Project.project\_id and tutor.tutor\_id=Project.tutor\_id

group by tutor.tutor\_id,Project.project\_id,postgraduate.postgraduate\_id,Project.allfunds

/\*指定学生参与项目的基本信息和确认情况\*//\*我们的ER图里没有确认情况的说明\*/

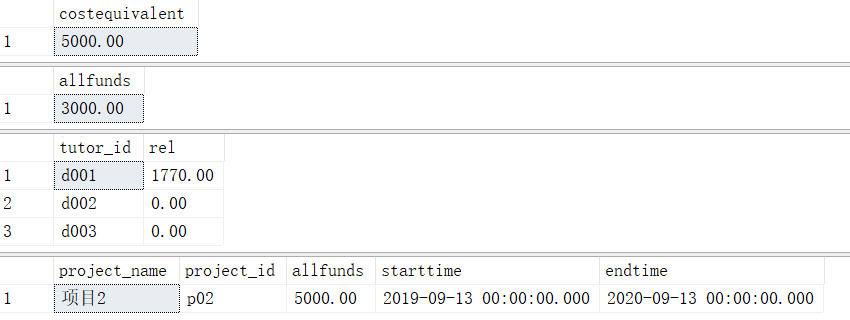
select distinct Project.project\_name,Project.project\_id,Project.allfunds,Project.starttime,Project.endtime

from Project,participate\_project,postgraduate\_participate\_project,tutor,postgraduate

where postgraduate.postgraduate\_id=postgraduate\_participate\_project.postgraduate\_id and participate\_project.project\_id=Project.project\_id

and postgraduate\_participate\_project.participate\_project\_id=participate\_project.participate\_project\_id and postgraduate\_name='yjs2'

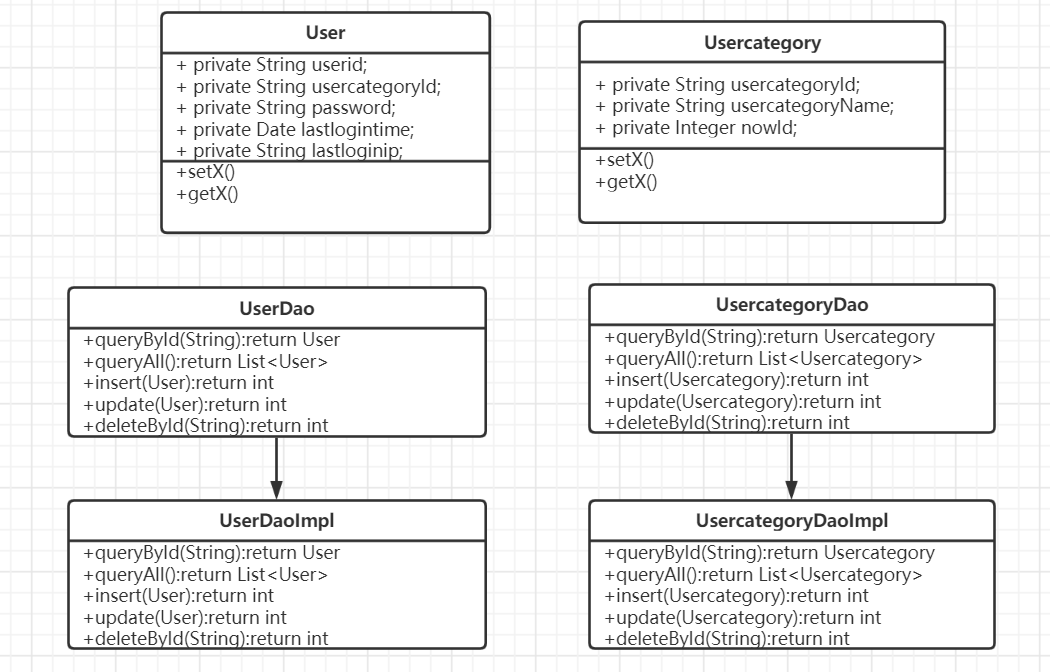
结果：满足要求



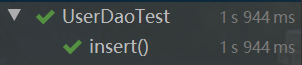
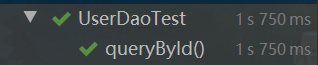
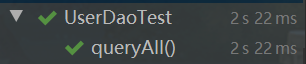
七、数据库持久层设计

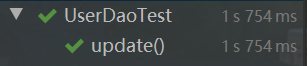
撰写要求：

* 列举具有关联关系的2个实体相关的UML图，含工厂，接口等。
* 描述列举的UML图中各类的API，包括类的名称，方法签名，方法概述。
* 描述编写的测试用例以及测试结果（只提供测试用例的名称，测试内容，测试是否通过即可，无需粘贴代码）。
* 小四，宋体，1.5倍行距。



测试用例测试了UserDao的增删改查功能，结果如下：





八、附件内容

提供内容：

* 根据课程设计产生的成果，列举内容，内容与提供的附件一一对顶，建议采用附件编号和附件名称的组织方式。
* 附件需要至少包含全局E-R图，数据库设计的DDL语句和持久层源代码。
* 附件还可包含其他表现工作量的材料。
* 提交报告前，删除报告中标注为黄色底色的模板信息。
* 小四，宋体，1.5倍行距。

附件：

1. 局部ER图：ks-part1.cdm,ks-part2.cdm,ks-part3.cdm
2. 整体ER图：中文：ks.cdm。英文：database.cdm
3. 数据库设计及数据填充：database\_english.sql
4. 持久层代码文件：Java.zip
5. 项目托管：https://github.com/Brief-drizzle/database-ks