

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的某项目的数据库设计 | | |
| 课程 | Oracle数据库应用 | | |
| 学 院 | 信息科学与工程学院 | | |
| 专 业 | 软件工程 | 年级 | 2018级 |
| 学生姓名 | 马芦 | 学号 | 201810414102 |
| 指导教师 | 赵卫东 | 职称 | 副教授 |

|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方 | 10 |  |
| 表设计 | 表，表空间设计合理，数据合理 | 20 |  |
| 用户管理 | 权限及用户分配方案设计正确 | 20 |  |
| PL/SQL设计 | 存储过程和函数设计正确 | 30 |  |
| 备份方案 | 备份方案设计正确 | 20 |  |
| **得分合计** | | |  |

2021 年 6 月 13日

目录

[图书管理系统数据库设计 3](#_Toc74509354)

[系统概述 3](#_Toc74509355)

[需求分析 3](#_Toc74509356)

[功能分析 3](#_Toc74509357)

[需求分析流程图 3](#_Toc74509358)

[数据库设计方案 5](#_Toc74509359)

[系统E-R图 5](#_Toc74509360)

[设计表 7](#_Toc74509361)

[物理模型设计图 10](#_Toc74509362)

[创建表空间 10](#_Toc74509363)

[实现表的创建及视图索引的设计 11](#_Toc74509364)

[触发器设计以及储存函数的设计 16](#_Toc74509365)

[角色分配以及权限赋予 19](#_Toc74509366)

[关于备份 21](#_Toc74509367)

# 图书管理系统数据库设计

## 系统概述

本系统是图书管理系统，图书管理是每个图书馆都要做的工作，一个良好的系统能够为管理图书的工作带来便利。本系统是图书管理系统的数据库设计模块，基于Oracle数据库。

## 需求分析

### 功能分析

1.学生可以直接通过借阅终端来查阅书籍信息﹐同时也可以查阅自己的借阅信息。

⒉当学生需要借阅书籍时，通过账号密码登陆借阅系统﹐借阅系统处理学生的借阅﹐同时修改图书馆保存的图书信息,修改被借阅的书籍是否还有剩余,同时更新学生个人的借阅信息。

3.学生借阅图书之前需要将自己的个人信息注册﹐登陆时对照学生信息。

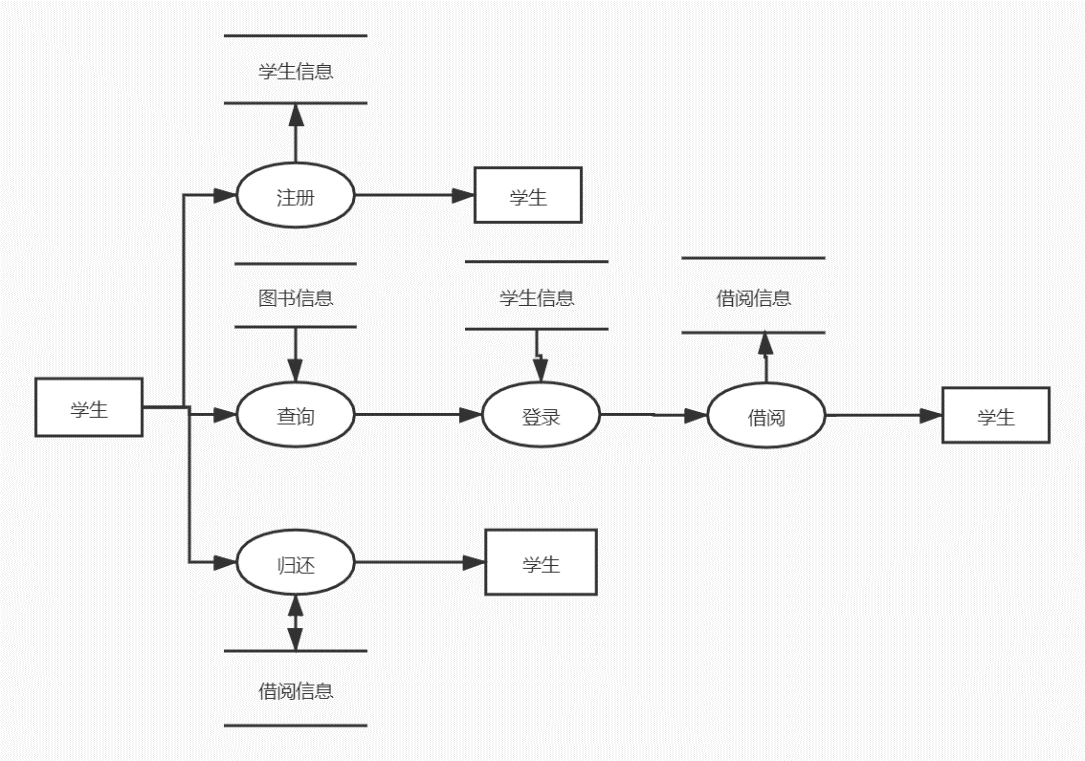
4.学生直接归还图书,根据图书编码修改借阅信息

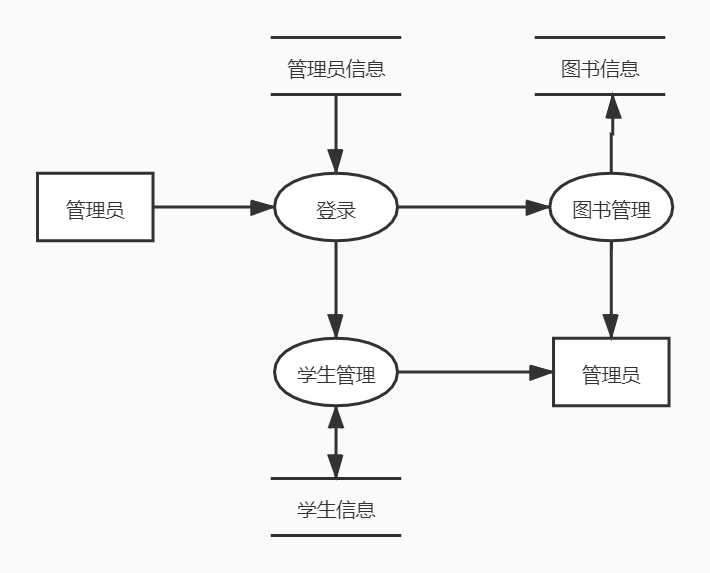
5.管理员登陆管理系统后，可以修改图书信息﹐增加或者删除图书信息

6.管理品可以注销学生信息。

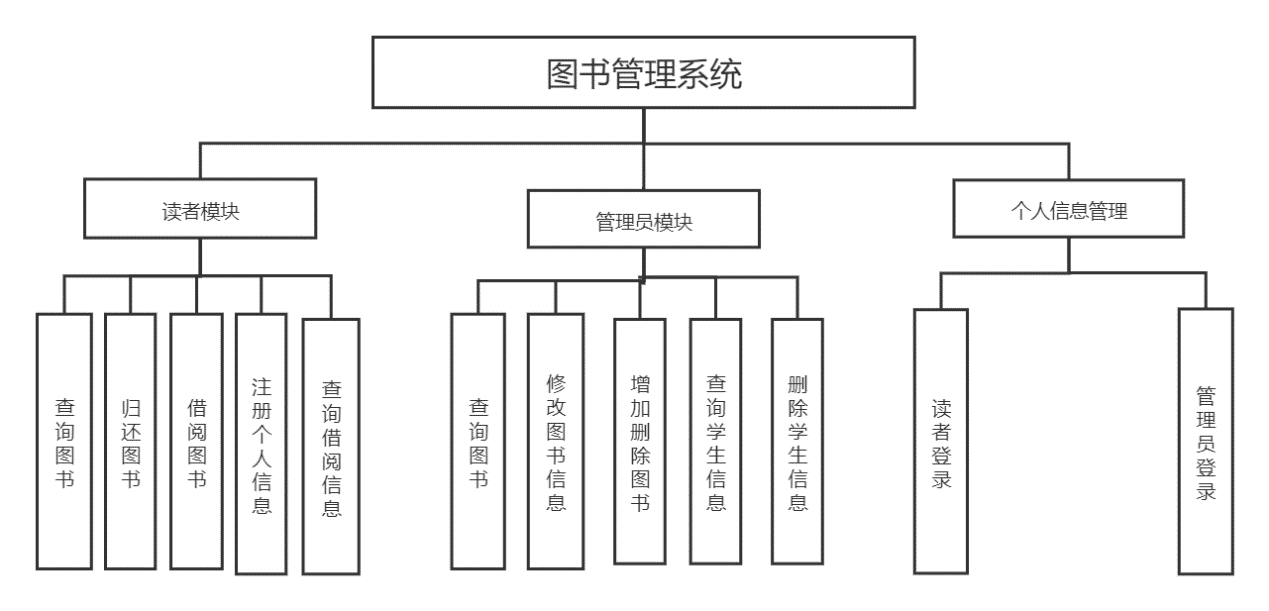
### 需求分析流程图

#### 数据流图





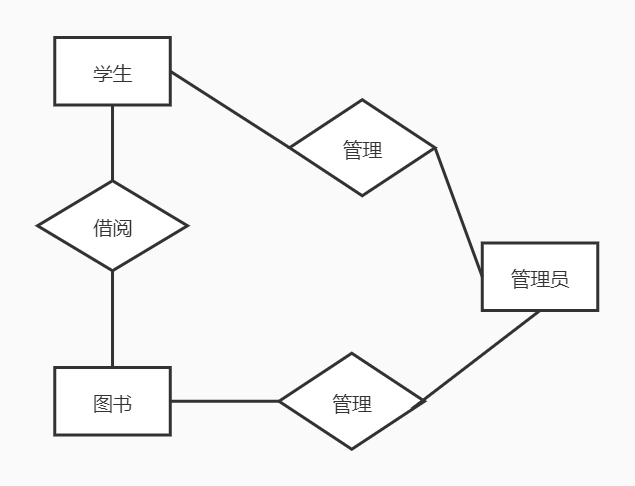
#### 系统功能模块图



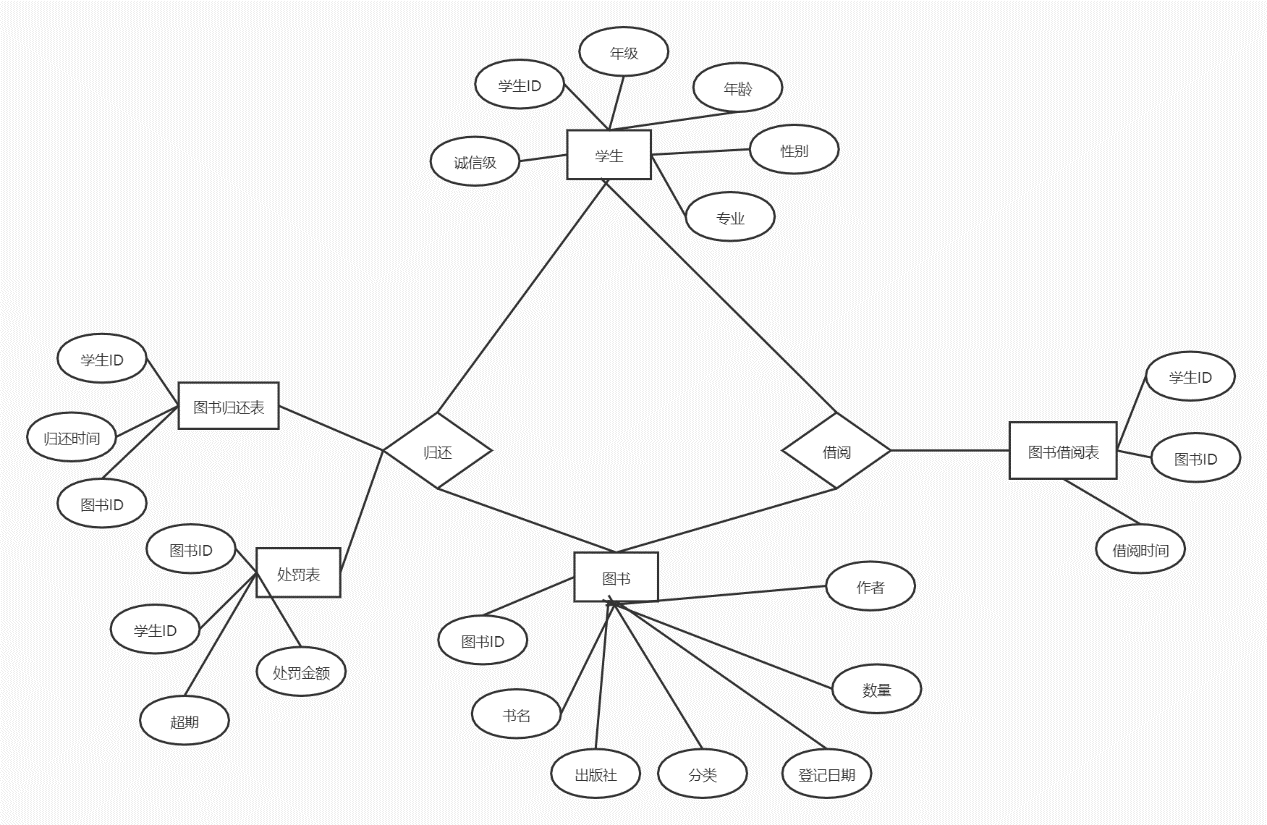
## 数据库设计方案

### 系统E-R图

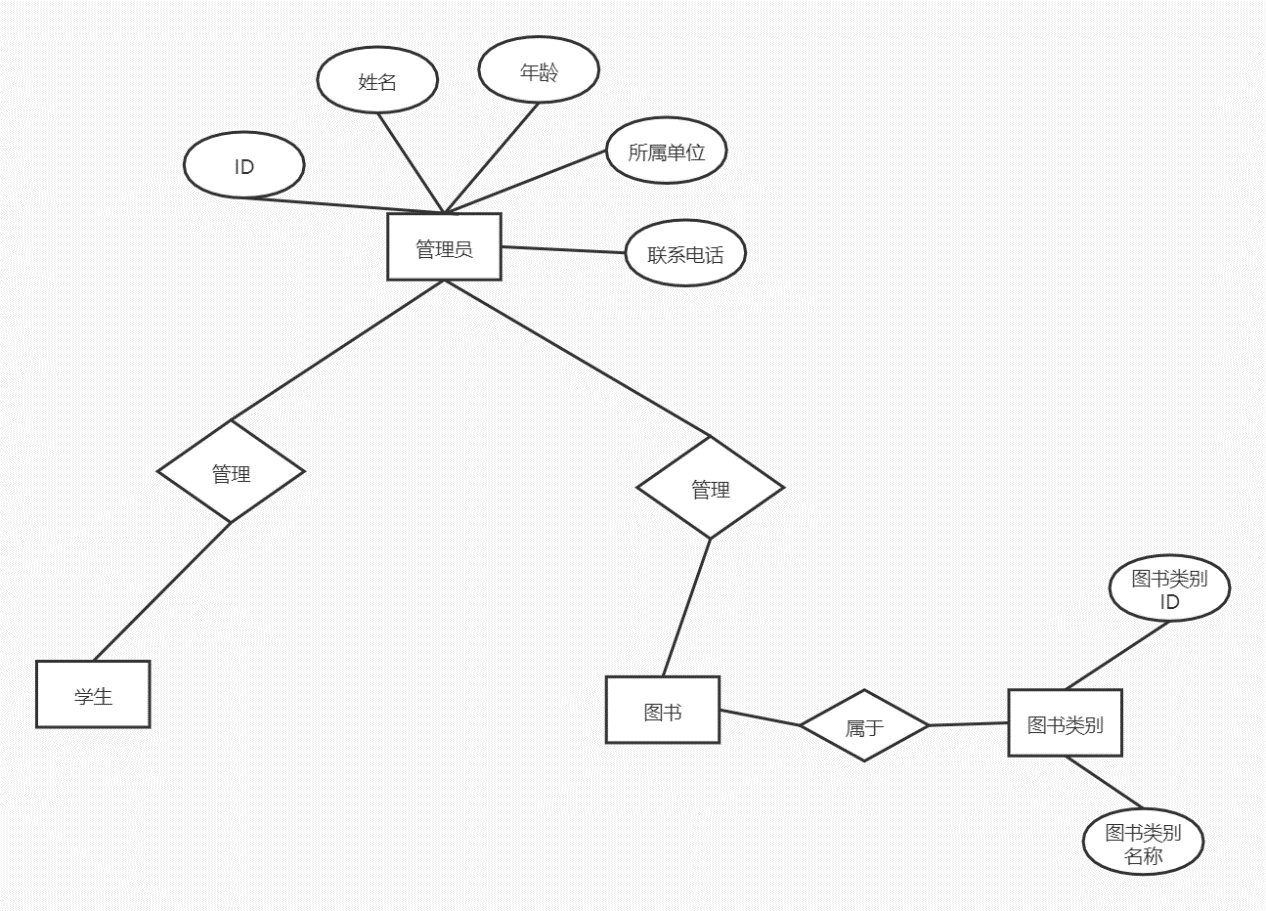
#### 总体E-R图



#### 读者E-R图



#### 管理员E-R图



### 设计表

#### Student表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| stu\_id | int | Not null/PK | 标明学生唯一学号 |
| stu\_name | varchar | Not null | 学生姓名 |
| stu\_gender | varchar | Not null | 学生性别 |
| stu\_age | int | Not null | 学生年龄 |
| stu\_pro | varchar | Not null | 学生专业 |
| stu\_grade | varchar | Not null | 学生年纪 |
| stu\_integrity | int | Not null/default=1 | 学生信用 |

#### Book表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| book\_id | int | Not null/PK | 唯一书籍序号 |
| book\_name | varchar | Not null | 书籍名称 |
| book\_author | varchar | Not null | 书籍作者 |
| book\_pub | varchar | Not null | 书籍出版社 |
| book\_num | int | Not null | 书籍状态 |
| book\_sort | varchar | Not null | 书籍分类 |
| book\_record | datetime | null | 书籍登记日期 |

#### book\_sort表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| sort\_id | varchar | Not null/PK | 类型编号 |
| sort\_name | varchar | Not null | 类型名称 |

#### borrow表储存学生借书信息

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| student\_id | varchar | Not null/PK | 学生编号 |
| book\_id | varchar | Not null/PK | 书籍编号 |
| borrow\_date | datetime | null | 借书时间 |
| expect\_return\_date | datetime | null | 预期归还时间 |

#### return\_table储存学生还书信息

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| student\_id | varchar | Not null/PK | 学生编号 |
| book\_id | varchar | Not null/PK | 书籍编号 |
| borrow\_date | datetime | null | 借书时间 |
| return\_date | datetime | null | 实际还书时间 |

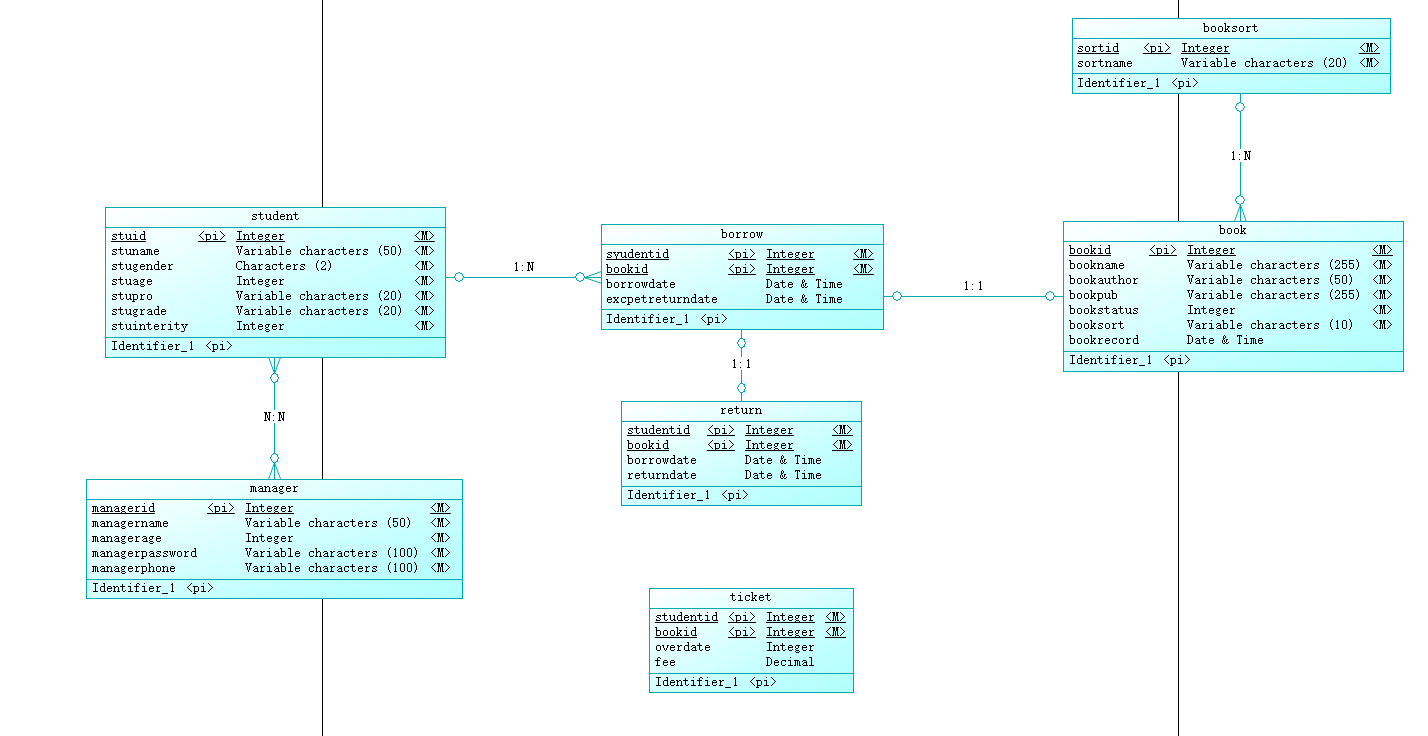
#### ticket储存学生罚单信息

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| student\_id | varchar | Not null/PK | 学生编号 |
| book\_id | varchar | Not null/PK | 书籍编号 |
| over\_date | int | null | 超期天数 |
| Ticket\_fee | float | null | 处罚金额 |

#### manager

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否为空/性质 | 说明 |
| manager\_id | varchar | Not null/PK | 管理员编号 |
| manager\_name | varchar | Not null | 管理员姓名 |
| manager\_age | varchar | Not null | 管理员年龄 |
| manager\_password | varchar | Not null | 管理员密码 |
| manager\_phone | varchar | Not null | 管理员电话 |

### 物理模型设计图



## 创建表空间

**第一个表空间**

Create Tablespace space\_qhl001

datafile

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_qhl001\_1.dbf'

SIZE 100M AUTOEXTEND ON NEXT 256M MAXSIZE UNLIMITED,

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_qhl001\_2.dbf'

SIZE 100M AUTOEXTEND ON NEXT 256M MAXSIZE UNLIMITED

EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO;

**第二个表空间**

Create Tablespace space\_qhl002

datafile

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_qhl002\_1.dbf'

SIZE 100M AUTOEXTEND ON NEXT 256M MAXSIZE UNLIMITED,

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_qhl002\_2.dbf'

SIZE 100M AUTOEXTEND ON NEXT 256M MAXSIZE UNLIMITED

EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO;

## 实现表的创建及视图索引的设计

#### 创建表

##### 创建读者表

CREATE TABLE "SCOTT"."READER"

(

  "STUDENT\_ID" INT PRIMARY KEY,

  "STUDENT\_NAME" VARCHAR(10) NOT NULL,

  "STUDENT\_GENDER" CHAR(2) NOT NULL,

  "STUDENT\_AGE" INT NOT NULL,

  " STUDENT\_PRO" VARCHAR(20) NOT NULL,

  " STUDENT\_GRADE" VARCHAR(20) NOT NULL,

  " STUDENT\_INTEGRITY" INT DEFAULT 1

)

TABLESPACE " qhl001";

ALTER TABLE "SCOTT"."READER" ADD CONSTRAINTS "Sex\_CK1" CHECK(Sex BETWEEN '男' AND '女');

##### 创建书籍表

CREATE TABLE "SCOTT"."BOOK"

(

  "BOOK\_ID" INT PRIMARY KEY,

  "BOOK\_NAME" VARCHAR(30) NOT NULL,

  "BOOK\_AUTHOR" VARCHAR(10) NOT NULL,

  "BOOK\_PUB" VARCHAR(40) NOT NULL,

  "BOOK\_NUM" INT NOT NULL,

  "BOOK\_SORT" VARCHAR(40) NOT NULL,

  " BOOK\_RECORD" DATETIME

)

TABLESPACE " qhl001";

##### 创建管理者表

CREATE TABLE "SCOTT"."MANAGER"

(

  "MANAGER\_ID" INT PRIMARY KEY,

  "MANAGER\_NAME" VARCHAR(10) NOT NULL,

  "MANAGER\_AGE" INT NOT NULL,

  " MANAGER\_PASSWORD" VARCHAR(20) NOT NULL,

  " MANAGER\_PHONE" VARCHAR(20) NOT NULL,)

TABLESPACE " qhl001";

##### 创建图书类别表

CREATE TABLE "SCOTT"."BOOK"

(

  "SORT\_ID" VARCHAR(10) PRIMARY KEY,

  "SORT\_NAME" VARCHAR(30) NOT NULLS

)

TABLESPACE " qhl001";

##### 创建借阅表

CREATE TABLE "SCOTT"."BORROW"

(

   "READER\_ID" INT REFERENCES "SCOTT"."READER"( READER\_ID),

   "BOOK\_ID" INT REFERENCES "SCOTT"."BOOK"( BOOK\_ID),

   "BORROW\_DATE" DATETIME,

   "EXPECT\_RETURN\_DATE" DATETIME

)

TABLESPACE " qhl002";

##### 创建归还表

CREATE TABLE "SCOTT"."RETURN"

(

   "READER\_ID" INT REFERENCES "SCOTT"."READER"( READER\_ID),

   "BOOK\_ID" INT REFERENCES "SCOTT"."BOOK"( BOOK\_ID),

   "BORROW\_DATE" DATETIME,

   "RETURN\_DATE" DATETIME

)

TABLESPACE " qhl002";

##### 创建罚单表

CREATE TABLE "SCOTT"."TICKET"

(

   "READER\_ID" INT REFERENCES "SCOTT"."READER"( READER\_ID),

   "BOOK\_ID" INT REFERENCES "SCOTT"."BOOK"( BOOK\_ID),

   "OVER\_DATE" DATETIME,

   "TICKET\_FEE" FLOAT

)

TABLESPACE " qhl002";

#### 创建索引

##### Student表创建索引

**1.为stu\_id创建索引，升序排序**

create index index\_id on student(stu\_id asc);

**2.为stu\_name 创建索引，并且降序排序**

alter table student add index index\_name(stu\_name, desc);

##### book表创建索引

**1.为book\_id创建索引，升序排列**

create index index\_bid on book(book\_id);

**2.为 book\_record创建索引，以便方便查询图书的登记日期信息，升序:**

create index index\_brecord on book(book\_record);

##### borrow表创建索引

**1.为stu\_id和book\_id创建多列索引:**

create index index\_sid\_bid on borrow(stu\_id asc, book\_id asc);

##### return表创建索引

1. **为stu\_id和book\_id创建多列索引∶**

create index index\_sid\_bid on return\_table(stu\_id asc, book\_id asc);

##### ticket表创建索引

1**.为stu\_id和book\_id创建多列索引:**

create index index\_sid\_bid on ticket(stu\_id asc, book\_id asc);

##### manager表创建索引

**1.为manager\_id创建索引:**

create index index\_mid on manager(manager\_id);

#### 创建视图

**1.读者视图在表student, borrow 和book 上创建借书者的全面信息视图stu\_borrow :**

Create force view “scott” ." stu\_borrow "

as

select student.stu\_id, book.book\_id, student.stu\_name, book.book\_name,

borrow\_date , adddate(borrow\_date,30) expect\_return\_date

from student, book, borrow

where student.stu\_id = borrow.stu\_id and book.book\_id = borrow.book\_id

when read only;

**2..创建个人所有借书归还纪录视图stu\_borrow\_return:**

Create force view “scott” ." stu\_borrow\_return"

as

select

student.stu\_id,

student.stu\_name,

book.book\_id,

book.book\_name,return\_table.borrow\_date,return\_table.return\_date

from student, book, return\_table

where student.stu\_id = return\_table.stu\_id and book.book\_id = return\_table.book\_id

with read only;

**3.创建读者的图书信息查询视图：**

create force view "scott"."v\_reader\_c1"

as

select "book\_id","book\_name","book\_author","book\_pub","book\_num" from "scott"."book"

with read only;

**4.管理者视图在表student, borrow 和book 上创建借书者的全面信息视图stu\_borrow :**

Create force view “scott” ." stu\_borrow "

as

select student.stu\_id, book.book\_id, student.stu\_name, book.book\_name,

borrow\_date , adddate(borrow\_date,30) expect\_return\_date

from student, book, borrow

where student.stu\_id = borrow.stu\_id and book.book\_id = borrow.book\_id

with check option;

**5.创建图书管理员的图书信息查询视图：**

create force view "scott"." "V\_Manager\_C1 "

as

select "book\_id","book\_name","book\_author","book\_pub","book\_num" from "scott"."book"

with check option;

## 触发器设计以及储存函数的设计

##### 触发器设计

1. **设计触发器borrow，当某学生借书成功后﹐图书表相应的图书不在架上,变为0∶**

create or replace trigger borrow

after insert on borrow

for each row

begin

update book set book\_num = book\_num -1

where book\_id = new.book\_id;

end borrow；

**⒉设计触发器trigger\_return，还书成功后﹐对应的书籍book\_num变为1:**

create or replace trigger trigger\_return

after insert on return\_table

for each row

begin

update book set book\_num = book\_num + 1

where book\_id = new.book\_id;

end trigger\_return;

**3.定义定时器（事件) eventJob ,每天自动触发一次﹐扫描视图stu\_borrow ，若发现当前有**

**预期归还时间小于当前时间,则判断为超期,生成处罚记录,这个定时器将每天定时触发存**

**储过程proc\_gen\_ticket :**

create event if not exists eventJob

on schedule every 1 DAYl\*每天触发\*\*

on completion PRESERVE

do call proc\_gen\_ticket(getdate());l\*调用存储过程\*/

set global event\_scheduler = 1;

alter event eventJob on completion preserve enable; \*开启定时器\*

**4.设计触发器trigger\_credit ，若处罚记录超过30条,则将这个学生的诚信级设置为0，下**

**次不允许借书∶**

create or replace trigger trigger\_credit

after insert on ticket

for each row

begin

if (select count(\*) from ticket where stu\_id=new.stu\_id)>30 then

update student set stu\_integrity = 0 where stu\_id = new.stu\_id;

end if;

end

##### 储存函数包设计

**创建程序包**

create or replace PACKAGE book\_package Is

function func\_get\_credit(stu\_id int) returns int;

function func\_get\_booknum(book\_id int) returns int;

procedure proc\_return(in stu\_id int, in book\_id int, in return\_date datetime);

procedure proc\_payoff(in stuid int, in bookid int);

procedure proc\_borrow(in stu\_id int,in book\_id int,in borrow\_date datetime);

procedure proc\_gen\_ticket(in currentdate datetime);

procedure stu\_register(in stu\_id int,in stu\_name varchar(20)，in stu\_sex varchar(20), in stu\_age int, in stu\_pro varchar(20), in stu\_grade varchar(20));

procedure ma\_register(in ma\_id int, in ma\_name varchar(20), in ma\_age int, in ma\_phone int);

end book\_package;

**创建储存过程**

create or replace PACKAGE body book\_package Is

/\* 1．设计存储过程，产生罚单proc\_gen\_ticket :

当日期超过预定归还日期时﹐产生罚单，并将记录写入表ticket中,这个存储过程在定时器

eventJob中调用: \*/

procedure proc\_gen\_ticket(in currentdate datetime)

BEGIN

declare cur\_date datetime;

set cur\_date = currentdate;

replace into ticket(stu\_id, book\_id, over\_date, ticket\_fee)

select

stu\_id,

book\_id,

datediff(cur\_date,stu\_borrow.expect\_return\_date),0.1\*datediff(cur\_date,stu\_borrow.exp

ect\_return\_date)

from stu\_borrow

where cur\_date>stu\_borrow.expect\_return\_date;

end proc\_gen\_ticket;

/\* ⒉设计学生注册信息存储过程∶学生注册信息stu\_register \*/

procedure stu\_register(in stu\_id int,in stu\_name varchar(20)，in stu\_sex

varchar(20), in stu\_age int, in stu\_pro varchar(20), in stu\_grade varchar(20))

begin

insert into student(stu\_id, stu\_name, stu\_sex,stu\_age, stu\_pro, stu\_grade)

values(stu\_id,stu\_name, stu\_sex,stu\_age,stu\_pro,stu\_grade);

end stu\_register;

/\* 3.设计管理员注册信息存储过程:ma\_register \*/

procedure ma\_register(in ma\_id int, in ma\_name varchar(20), in ma\_age int, in

ma\_phone int)

BEGIN

insert into manager

values(ma\_id, ma\_name, ma\_age, ma\_phone);

END ma\_register;

/\* 4．借书过程的实现:1)设计存储函数, func\_get\_credit ,返回学生的诚信级︰\*/

function func\_get\_credit(stu\_id int) returns int

begin

return(select stu\_integrity from student where student.stu\_id = stu\_id);

end

/\* 2)设计存储函数，func\_get\_booknum ,返回书籍是否在架上:\*/

function func\_get\_booknum(book\_id int) returns int

begin

return(select book\_num from book where book.book\_id = book\_id);

end

/\*3)设计存储过程proc\_borrow ,调用func\_get\_credit和 func\_get\_booknum ,判断这

个学生诚信度和书籍是否在架上,若为真,则借书成功,在borrrow表中插入纪录;

否则提示失败︰\*/

procedure proc\_borrow(in stu\_id int,in book\_id int,in borrow\_date datetime)

begin

if func\_get\_credit(stu\_id)= 1 and func\_get\_booknum(book\_id)= 1 then

insert into borrow

values(stu\_id, book\_id, borrow\_date);

else

select 'failed to borrow';

end if;

end proc\_borrow;

/\* 5.还书存储过程proc\_return :

当还书时,查看是否书是否超期,即查询ticket表项,当发现超期,提示交罚单后再次还书 ，

如没有超期，则纪录归还项目到return\_table中，并且删除借书纪录(以免还书后定时器仍

然扫描这个纪录): \*/

procedure proc\_return(in stu\_id int, in book\_id int, in return\_date datetime)

begin

DECLARE borrowdate datetime;

if(select payoff from ticket where ticket.stu\_id =stu\_idand

ticket.book\_id=book\_id)= 1 then l\*判断是否交了罚单﹐1表示没有交\*

select 'please pay off the ticket';

elsel\*纪录归还项目到return\_table 中，并且删除借书纪录\*/

set borrowdate = (select borrow\_date from borrow where borrow.stu\_id =

stu\_id and borrow.book\_id = book\_id);

insert into return\_table

values(stu\_id, book\_id, borrowdate, return\_date);

delete from borrow

where borrow.stu\_id = stu\_id and borrow.book\_id = book\_id;

end if;

end proc\_return;

/\* 6.交罚单存储过程∶修改罚单中payoff 段为0，表明罚单已交∶ \*/

procedure proc\_payoff(in stuid int, in bookid int)

begin

update ticket

set payoff = o

where ticket.stu\_id = stuid and ticket.book\_id = bookid;

select 'succeed';

end proc\_payoff;

end book\_package;

## 角色分配以及权限赋予

**创建学生用户：**

create user student1

identified by 123

default tablespace users

quota 50M on users

password expire;

create user student2

identified by 123

default tablespace users

quota 50M on users

password expire;

**创建图书管理员用户**：

create user manager1

identified by 123

default tablespace users

quota 50M on users

password expire;

 create user manager2

identified by 123

default tablespace users

quota 50M on users

password expire;

**创建角色：**

create role student

not identified;

create role manager

not identified;

**将角色赋予用户：**

GRANT CONNECT,student TO student1;

GRANT CONNECT,student TO student2;

GRANT CONNECT,manager TO manager1;

GRANT CONNECT,manager TO manager2

**为角色赋予权限：**

grant select on "scott"."v\_reader\_c1" to student;

grant select on "scott"."v\_reader\_c2" to student;

grant select on "scott"."stu\_borrow” to student;

grant select on "scott"." stu\_borrow\_return” to student;

grant connect,resource,execute to manager;

grant select,insert,delete,update on "scott"."v\_manager\_c1" to manager;

grant select,insert,delete,update on "scott"." stu\_borrow " to manager;

## 关于备份

一、原理

写一个数据库备份、删除脚本，加入到windows任务计划里，每天定时执行即可；

二、步骤

1. 新建txt文档，输入如下“oracle数据库定时备份、删除脚本”内容
2. 脚本内容如下：

@echo off

echo ================================================

echo Windows环境下Oracle数据库的自动备份脚本

echo 1. 使用当前日期命名备份文件。

echo 2. 自动删除7天前的备份。

echo ================================================

::以“YYYYMMDD”格式取出当前时间。

set BACKUPDATE=%date:~0,4%%date:~5,2%%date:~8,2%

::设置用户名、密码和要备份的数据库。

set USERA=sccot

set PASSWORD=123456

set SID=orcl

::创建备份目录。

if not exist "C:\oraclebackup\data" mkdir C:\oraclebackup\data

if not exist "C:\oraclebackup\log" mkdir C:\oraclebackup\log

set DATADIR=C:\oraclebackup\data

set LOGDIR=C:\oraclebackup\log

exp %USERA%/%PASSWORD%@%SID% file=%DATADIR%\%BACKUPDATE%.dmp

::删除7天前的备份。

forfiles /p "%DATADIR%" /s /m \*.\* /d -7 /c "cmd /c del @path"

forfiles /p "%LOGDIR%" /s /m \*.\* /d -7 /c "cmd /c del @path"

exit

3另存为.bat格式

4.windows定时添加任务