# 高校招生系统说明书

## 需求分析：系统开发背景分析

### 背景

高等院校的招生录取工作，是一个复杂工作。涉及到大量考生和院校的数据，人工进行录取工作量大、主观性强、容易滋生腐败行为。通过开发“招生录取系统”，实现录取过程的自动化等工作。大大节约录取时间、使录取更加公正、客观。

### 功能特性

（蓝色标出已实现特性）

FE1.高校招生计划录入

FE2.学生志愿录入

FE3.各学生信息一览

FE4.自动投档

FE5.高校招生列表查询

FE6.学生录取结果查询

FE7.各高校招生情况一览

FE8.一键清空招生信息

### 可选功能

AFE1.高校招生结果报表打印

AFE2.学生录取结果报表打印

AFE3.学生成绩修改跟踪

AFE4.学生录取结果修改跟踪

AFE5.高校调剂意向录入

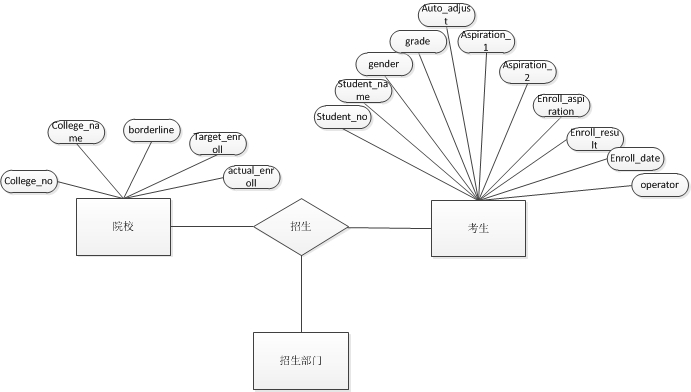
AFE6.学生调剂意向录入

AFE7.自动调剂

AFE8.一键清空调剂结果

## 系统分析与设计：重点为数据库表结构设计，各表的描述等。

### 实体关系



### 表结构设计

高校(高校代码,高校名称,录取分数线,预计录取数)

学生(准考证号,学生姓名,性别,高考成绩)

志愿(学生准考证号,志愿优先级,院校代码)

调剂意愿(学生准考证号,是否调剂)

-- 0未录取,-1调剂录取,1第一志愿录取,2第二志愿录取

录取结果(学生准考证号,录取结果)

-- -1退档,非-1院校代码

调剂结果(学生准考证号,调剂结果)

## 系统实现：主要包含数据库用户的创建、创建表的脚本、存储过程及触发器实现脚本等。

### 创建用户并授权

|  |
| --- |
| CREATE USER yjy\_homework IDENTIFIED BY 111111 default tablespace MY\_TABLESPACE; |
| GRANT CONNECT,RESOURCE TO yjy\_homework;  GRANT CREATE TABLE,CREATE ANY INDEX,CREATE VIEW,CREATE SEQUENCE,CREATE SYNONYM TO yjy\_homework; |
| GRANT UNLIMITED TABLESPACE TO yjy\_homework; |

### 表、序列、索引

#### 表

|  |  |
| --- | --- |
| 高校表 | CREATE TABLE college (  college\_number NUMBER(4) PRIMARY KEY,  college\_name VARCHAR2(30) NOT NULL,  borderline NUMBER(3) CHECK (borderline BETWEEN 300 AND 700),  targetEnroll NUMBER(3) CHECK (targetEnroll <= 100),  actualEnroll NUMBER(3) DEFAULT 0  ) |
| 学生表 | CREATE TABLE STUDENT (  student\_number NUMBER(5) PRIMARY KEY,  student\_name VARCHAR2(15) NOT NULL,  gender NCHAR(1) CHECK (gender IN ('男', '女')),  grade NUMBER(3) CHECK (grade BETWEEN 0 AND 700)  ) |
| 学生志愿表 | CREATE TABLE student\_will(  student\_number NUMBER(5) REFERENCES STUDENT,  priority NUMBER(2),  aspiration NUMBER(4) REFERENCES COLLEGE,  PRIMARY KEY (student\_number,priority) |
| 调剂意愿表 | CREATE TABLE auto\_adjust\_will (  student\_number NUMBER(5) REFERENCES STUDENT,  auto\_adjust NUMBER(1) DEFAULT 0  ) |
| 录取结果表 | CREATE TABLE student\_enroll\_result(  student\_number NUMBER(5) PRIMARY KEY REFERENCES STUDENT,  enroll\_result NUMBER(1) -- 0未录取,-1调剂录取,1第一志愿录取,2第二志愿录取  "ENROLL\_OPERATOR" NVARCHAR2(15) DEFAULT NULL  ) |
| 调剂录取结果表 | CREATE TABLE auto\_adjust\_result(  student\_number NUMBER(5) PRIMARY KEY REFERENCES STUDENT,  college\_number NUMBER(5) REFERENCES COLLEGE  ) |

#### 序列

|  |  |
| --- | --- |
| 学生表id生成序列 | CREATE SEQUENCE student\_sequence START WITH 1 |
| 高校表id生成序列 | CREATE SEQUENCE college\_sequence START WITH 1 |

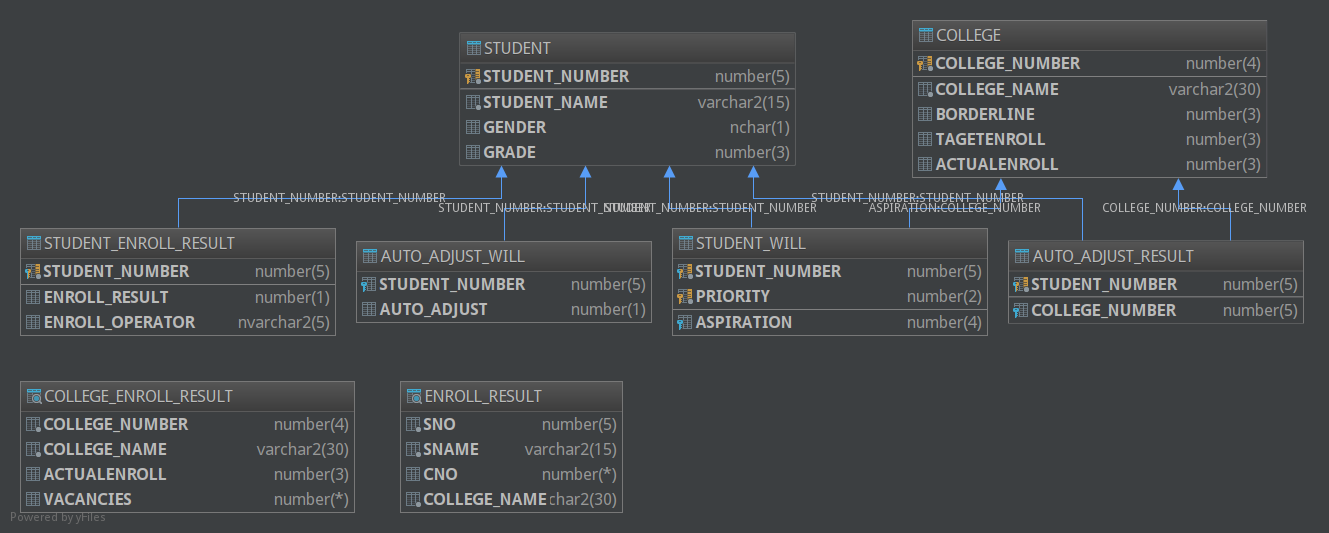
#### 索引

暂不需要。

### 视图

|  |  |
| --- | --- |
| 考生录取结果(考生编号，姓名，院校代码，院校名称)  （包括志愿录取、调剂录取和退档的所有考生） | CREATE OR REPLACE VIEW ENROLL\_RESULT AS  SELECT  sno,  sname,  cno,  COLLEGE\_NAME  FROM (  SELECT  STUDENT\_NUMBER sno,  STUDENT\_NAME sname,  FIND\_ENROLL\_COLLEGE(STUDENT\_NUMBER) cno  FROM STUDENT  ) my\_table, COLLEGE  WHERE my\_table.cno = COLLEGE\_NUMBER; |
| 录取情况(院校编号，院校名称，招生人数，缺额) | CREATE OR REPLACE VIEW COLLEGE\_ENROLL\_RESULT AS  (  SELECT  COLLEGE\_NUMBER,  COLLEGE\_NAME,  ACTUALENROLL,  TARGETENROLL - ACTUALENROLL Vacancies  FROM (SELECT  COLLEGE\_NUMBER,  COLLEGE\_NAME,  TARGETENROLL,  COUNT\_ACTUAL\_ENROLL(COLLEGE\_NUMBER) ACTUALENROLL  FROM COLLEGE)  ) |
| 志愿视图(学生编号,志愿1,志愿1学校名称,志愿2,志愿2学校名称,是否调剂) | CREATE OR REPLACE VIEW ALL\_WILL AS  SELECT  STUDENT\_NUMBER,  ASPIRATION1,  FIND\_COLLEGE\_NAME(ASPIRATION1) CNAME1,  ASPIRATION2,  FIND\_COLLEGE\_NAME(ASPIRATION2) CNAME2,  AUTO\_ADJUST  FROM (  SELECT  STUDENT\_NUMBER,  GET\_ASPIRATION(STUDENT\_NUMBER, 1) ASPIRATION1,  GET\_ASPIRATION(STUDENT\_NUMBER, 2) ASPIRATION2,  AUTO\_ADJUST  FROM AUTO\_ADJUST\_WILL  ) |

表、视图间关系如图：



### 初始化数据

数据已导出为.sql文件，放置在src/main/doc/testdata目录下。

### 函数、存储过程、触发器、pl/sql

#### 函数

|  |  |
| --- | --- |
| 根据学生编号找出考入的高校编号，退档则为null | create FUNCTION find\_Enroll\_College(sno NUMBER) RETURN NUMBER  IS  college\_number NUMBER(5);  v\_enroll\_result NUMBER(1);  BEGIN  SELECT ENROLL\_RESULT INTO v\_enroll\_result FROM STUDENT\_ENROLL\_RESULT WHERE STUDENT\_NUMBER=sno;  IF (v\_enroll\_result=0) THEN -- 退档  SELECT NULL INTO college\_number FROM dual;  ELSIF (v\_enroll\_result=-1) THEN -- 调剂录取  SELECT COLLEGE\_NUMBER INTO college\_number FROM AUTO\_ADJUST\_RESULT WHERE AUTO\_ADJUST\_RESULT.STUDENT\_NUMBER=sno;  ELSIF (v\_enroll\_result=1) THEN -- 第一志愿录取  SELECT ASPIRATION INTO college\_number FROM STUDENT\_WILL WHERE STUDENT\_WILL.STUDENT\_NUMBER=sno AND STUDENT\_WILL.PRIORITY=1;  ELSIF (v\_enroll\_result=2) THEN -- 第二志愿录取  SELECT ASPIRATION INTO college\_number FROM STUDENT\_WILL WHERE STUDENT\_WILL.STUDENT\_NUMBER=sno AND STUDENT\_WILL.PRIORITY=2;  ELSE -- 异常  SELECT NULL INTO college\_number FROM dual;  END IF;  RETURN college\_number;  END; |
| 计算高校的实际招生人数 | CREATE FUNCTION count\_actual\_enroll(p\_college\_number NUMBER)  RETURN NUMBER  IS  enroll\_count NUMBER;  BEGIN  SELECT count(\*)  INTO enroll\_count  FROM ENROLL\_RESULT  WHERE ENROLL\_RESULT.CNO = p\_college\_number;  RETURN enroll\_count;  END; |
| 根据学生编号和志愿号得出志愿高校代码 | CREATE OR REPLACE FUNCTION GET\_ASPIRATION(sno IN NUMBER,ano IN NUMBER) RETURN NUMBER  AS  college NUMBER;  BEGIN  SELECT ASPIRATION INTO college FROM STUDENT\_WILL WHERE STUDENT\_NUMBER=sno AND PRIORITY=ano;  RETURN college;  END; |
| 根据高校编号选出高校名称 | CREATE OR REPLACE FUNCTION FIND\_COLLEGE\_NAME(cno IN NUMBER) RETURN NVARCHAR2  AS  cname NVARCHAR2;  BEGIN  SELECT COLLEGE\_NAME INTO cname FROM COLLEGE WHERE COLLEGE\_NUMBER=cno;  RETURN cname;  END; |
| 判断某个学生是否已被录取 | create OR REPLACE FUNCTION IS\_STUDENT\_ENROLLED(sno IN NUMBER)  RETURN NUMBER  AS  is\_enrolled NUMBER(1);  BEGIN  SELECT count(\*)  INTO is\_enrolled  FROM STUDENT\_ENROLL\_RESULT  WHERE STUDENT\_NUMBER = sno AND ENROLL\_RESULT IS NOT NULL AND ENROLL\_RESULT != 0;  RETURN is\_enrolled;  END; |

#### 存储过程

|  |  |
| --- | --- |
| 清空录取结果 | create OR REPLACE PROCEDURE clear\_status  AS  BEGIN  DELETE STUDENT\_ENROLL\_RESULT;  DELETE AUTO\_ADJUST\_RESULT;  COMMIT ;  END; |
| 投档(高校编号,志愿编号) | CREATE OR REPLACE PROCEDURE ENROLL\_PROC(P\_YXBH IN NUMBER, WILL\_NUMBER IN NUMBER)  AS  V\_ZSRS NUMBER(3); -- 招生人数  V\_LQFSX NUMBER(3); -- 录取分数线  V\_YXMC VARCHAR2(30); -- 院校名称  V\_COUNT NUMBER(3);  BEGIN  SELECT  --取院校信息  TARGETENROLL,  BORDERLINE,  COLLEGE\_NAME  INTO V\_ZSRS, V\_LQFSX, V\_YXMC  FROM COLLEGE  WHERE COLLEGE\_NUMBER = P\_YXBH;  V\_COUNT := 0;  -- 志愿为高校P\_YXBH,且成绩高于分数线,且尚未被录取的学生  FOR STU\_REC IN (SELECT STUDENT.STUDENT\_NUMBER  FROM STUDENT, STUDENT\_WILL  WHERE IS\_STUDENT\_ENROLLED(STUDENT.STUDENT\_NUMBER)=0  AND PRIORITY = WILL\_NUMBER  AND ASPIRATION = P\_YXBH  AND STUDENT.STUDENT\_NUMBER = STUDENT\_WILL.STUDENT\_NUMBER  AND GRADE >= V\_LQFSX  ORDER BY GRADE DESC) LOOP  EXIT WHEN V\_COUNT >= V\_ZSRS;  -- 插入录取结果  INSERT INTO STUDENT\_ENROLL\_RESULT VALUES  (STU\_REC.STUDENT\_NUMBER, WILL\_NUMBER, USER);  V\_COUNT := V\_COUNT + 1;  END LOOP;  COMMIT;  END; |
| 一键自动投档 | CREATE OR REPLACE PROCEDURE AUTO\_PROC  AS  v\_max\_priority NUMBER(2);  v\_min\_priority NUMBER(2);  BEGIN  FOR COLLEGE\_REC IN (SELECT COLLEGE\_NUMBER  FROM COLLEGE) LOOP  SELECT  max(PRIORITY),  min(PRIORITY)  INTO v\_max\_priority, v\_min\_priority  FROM STUDENT\_WILL;  -- 第n志愿录取  FOR n IN v\_min\_priority..v\_max\_priority LOOP  ENROLL\_PROC(COLLEGE\_REC.COLLEGE\_NUMBER, n);  END LOOP;  END LOOP;  END; |
|  | -- 尚未被录取，且同意调剂的学生  CURSOR adjustable\_student IS  SELECT STUDENT\_NUMBER  FROM AUTO\_ADJUST\_WILL  GROUP BY STUDENT\_NUMBER  HAVING IS\_STUDENT\_ENROLLED(STUDENT\_NUMBER) = 0;  -- 调剂录取  FOR v\_student IN adjustable\_student LOOP  SELECT \*  INTO v\_college\_number  FROM COLLEGE, COLLEGE\_ENROLL\_RESULT, STUDENT  WHERE VACANCIES > 0 AND COLLEGE.COLLEGE\_NUMBER = COLLEGE\_ENROLL\_RESULT.COLLEGE\_NUMBER AND  STUDENT\_NUMBER = v\_student.STUDENT\_NUMBER AND GRADE > BORDERLINE;  END LOOP;  V\_COUNT := V\_COUNT + 1;  COMMIT; |

#### 触发器

|  |  |  |
| --- | --- | --- |
| 已删除，采用视图提取高校实际招生人数，而不直接将数据保存在基表。因为oracle没有计算列，触发器又因为读锁而无法计算。 | 变动志愿录取结果表时自动更新学校实际录取人数（也可以将学校实际录取人数做成计算列） | CREATE OR REPLACE TRIGGER UPDATE\_COLLEGE\_ENROLL  AFTER INSERT OR UPDATE OR DELETE  ON STUDENT\_ENROLL\_RESULT  BEGIN  UPDATE COLLEGE  SET ACTUALENROLL = COUNT\_ACTUAL\_ENROLL(COLLEGE\_NUMBER);  END; |
| 变动调剂录取结果表时自动更新学校实际录取人数（也可以将学校实际录取人数做成计算列） | CREATE OR REPLACE TRIGGER UPDATE\_COLLEGE\_ENROLL\_ADJUST  AFTER INSERT OR UPDATE OR DELETE  ON AUTO\_ADJUST\_RESULT  BEGIN  UPDATE COLLEGE  SET ACTUALENROLL = COUNT\_ACTUAL\_ENROLL(COLLEGE\_NUMBER);  END; |

#### pl/sql

|  |  |
| --- | --- |
| 创建测试数据 | DECLARE  sno NUMBER(5);  gender NVARCHAR2(1);  BEGIN  sno := STUDENT\_SEQUENCE.nextval;  IF (mod(sno,2) = 1)  THEN  gender := '男';  ELSE  gender := '女';  END IF;  INSERT INTO STUDENT VALUES (sno, '学生' || sno, gender, 500 + sno);  Commit；  END; |

### 权限与安全

暂无。

### 性能分析

|  |  |
| --- | --- |
| 查询 | 视图的定义、以及所有的查询语句，其中用到的查询连接都是基于主键的连接。按照当前的表结构设计，即使不建索引查询速度也不会慢多少。 |
| 并发 | 学生和高校信息录入功能暂不开放，因此不考虑这两者相关的并发情况；一键录取功能是基于ENROLL\_PROC存储过程实现的，经过测试，该存储过程能完全保障数据一致性。 |

## 备份方案

暂无。

## 心得体会

①dml触发器中不能调用读该表的函数，因为此时表发生了变化, 触发器/函数不能读它。

②数据库设计时不要考虑过多扩展性，这会导致编码难度大大增加（增删改都要专门编写一个存储过程）。按照ORM的思路建表最合适，最终结果大概符合2NF或3NF，但是存取数据的sql可以很简单地编写，也和ORM框架有很好的契合。