Student Registration System

Using PL/SQL and JDBC

For CS 532 (2019)

Prepared by:
Jasmeet Kaur Dua, Harshit Vadodaria

Table of Contents

1. INTRODUCTION

1.1 Purpose

This software design specification is made with the purpose of comprehensive design of the software architecture and system in detail. The document will provide all the details which help to understand the functioning of the design more effectively. Moreover, the document facilitates communication and understanding of the system
by providing several views of the system design.

1.2 Scope

Every semester, hundreds of students got the admission in the university to keep the track of every student for every semester is a challenging task, using the student registration system helps to organize it in a better way.
For each semester multiple new students enrolled into the university, then the process of keeping the track of new students, enrollments the courses while checking the all the details i.e. whether the prerequisite have already completed by the student or not with the minimum grade to qualify, including the details of the classes i.e. limit of the class, in which section students is enrolled, in which year students is enrolling for which all classes, the maximum number of classes in which student can be enrolled and many more criteria we have covered on which students registration system will be used to optimize the whole procedure.

1.3 Key Findings

There are two distinct situations in which a student may be registered with a school: when enrolling for the first time and when allocating the classes for each period.
Our implementation design will cover all the details related to the all the procedures and functions which briefly includes:

1. Adding a new student
2. Delete a student
3. List of courses and their prerequisite
4. Enrolling students in courses
   Multiple constrains while enrolling the students as:
   - Checking whether student is already enrolled in the same class
   - Limit on opting a course for a student
   - Checks for a valid student id while opting for any courses
   - Checks whether student is enrolling for a valid course or not
   - Prerequisite must be complete before enrolling in any course
5. Dropping a student from a class
6. List the courses and classes a student is currently enrolled in
7. List the details of students enrolled in a particular class
8. List Prerequisite courses of a course
9. Listing the tuples in every table

2. IMPLEMENTATION DESIGN

2.1 Procedures

A procedure is a named PL/SQL subprogram which can (optionally) accept parameters and may or may not return a value to the host. Its major function is to embed a business logic process and perform data manipulation with the help of the supplied data.

A PL/SQL procedure is a reusable unit that encapsulates specific business logic of the application. Technically speaking, a PL/SQL procedure is a named block stored as a schema object in the Oracle Database

For the implementation of SRS (Student Register System) we created multiple procedures as per the requirements, below mentioned the procedures we used in database to manipulate the data (most of the procedures are self-explainatory):

- show_students
- show_courses
- show_classes
- show_enrollment
- show_prerequisites
- show_logs
- add_student
- delete_student
- show_prerequisites (of a specific course)
- show_enrollment_details (lists the courses a student is enrolled in)
- show_class_students (list the students enrolled in a specific class)
- enroll_students (enrolls a student into a class)
- drop_class (drops a student from a class)

2.2 Functions

Functions is a standalone PL/SQL subprogram. Like PL/SQL procedure, functions have a unique name by which it can be referred. These are stored as PL/SQL database objects.
A stored function (also called a user function or user-defined function) is a set of PL/SQL statements you can call by name. Stored functions are very similar to procedures, except that a function returns a value to the environment in which it is called. User functions can be used as part of a SQL expression.

Here we used the functions for showing the desired output to the user on user interface, below are the functions we define in our database (all these functions server the same purpose as the above procedures with similar names, except, these functions return an instance of ref cursor, so data can be referenced and fetched in the Java Application):

- get_students
- get_courses
- get_classes
- get_enrollments
- get_prerequisites
- get_logs
- get_enrollment_details
- get_class_students (gets student enrolled in a particular class)
- get_prerequisites (of a specific course)

a) get_students: get_students function use to fetch the details of all the students in the university, which includes student id, first name, last name, status ('freshman', 'sophomore', 'junior', 'senior', 'graduate'), gpa and email.

b) get_courses: get_courses function use to fetch the details of all the courses provided by the university which includes department code, course number and title.

c) get_classes: before enrolling in any courses it is necessary to check all the details of the classes available such as sections available for the particular semester of that year, limit and size of the class for that course.

d) get_enrollments: by calling this procedure one can check which student enrolled in which course and their grades ('A', 'B', 'C', 'D', 'F', 'I').

e) get_prerequisites: by choosing this procedure a student can check all the prerequisite related to any course before opting any course.

f) get_logs: all the actions like insert, delete, update is store in the log table, all details can be fetched by using this function.

g) get_enrollment_details: for student specific enrollment details, get_enrollment_details function can be used. By passing the student id in this function, all the courses in which that student has enrolled will be on display.

h) get_class_students: there are multiple students who enrolled in a single class, to fetch the details of all the students this function needs to be call. It can be checked by passing the department code along with the class number to check the students who have enrolled for that class.

3. ARCHITECTURE DESIGN

- MVC Architecture Style (Model – View – Controller)

  MVC Style separates presentation and interaction from the system data. The system is structured into three logical components that interact with each other.

  1. The Model component – Comprises of the underlying Database engine and the state of the data stored in the database (in this case, Oracle DB on Bingsuns Server)
  2. The View component- Defines the strategy to present the data to the user (in this case, Java Std In/Std Out Console)
  3. The Controller component- Manages user interaction and acts as intermediate between the View and the Model, communicating data between them. (In this case, the Java Application that uses JDBC driver and Java SQL Libraries)

  We will use this MVC Style for the Student Registration System because, there are multiple ways to view and interact with data. Here we are using Java as a controller component.

4. USER INTERFACE DESIGN

The Application is a Console-based, menu-driven application.
The Console User interface allows the user to choose the options from the provided menu, it helps user to choose a course of action that needs to be executed using the console. From the java console user can choose the options which further ask for the required attributes and display the result based on the selection.

4.1 Console

The Console prompts the user to select the action from the set of menu options, throws error if the selection is invalid.
Promptly executes the selected task if the correct input will be given.
For each task, the user will see some output, either in the form of the data expected to be shown, or a status text (ex. Completed successfully).

```
******* Student Registration System *******
Choose any of the below options
1.      Show all data from tables
2.      Create new/Delete existing Student
3.      Show Prerequisites of a course
4.      Show student's enrollment details
5.      Modify a student's enrollment status for a class
6.      Show students enrolled in a class
0.      Exit
```

1: Main Menu of the Application Console

4.2 Menu Options

The menu includes the below options:

1. Show all data from tables
   - Selecting this option would open up another menu (sub-menu/secondary menu) which would let the user choose from the names of available tables in the database, following which, all the tuples from that table would be listed
   - Also includes Option to go back to the main menu

2. Create new/Delete existing student
   - Opens up another sub-menu, that lets the user choose between adding a new student (the user will be prompted for all inputs, including sid, firstname, etc.), and deleting an existing student (user will be prompted for sid; Console shows 'sid not found' for invalid sid)
   - Also includes Option to go back to the main menu

3. Show a student's enrollment details
   - Prompts the user for an sid, and then lists the classes the student is enrolled in (after performing validation on sid)

4. Show prerequisites of a course
   - Prompts the user for a Department Code (ex. CS), followed by a Course Number within that department (ex. 532), and then lists all the direct and indirect prerequisites of the entered course.
   - There is no limit to the number of levels of checking prerequisites, which means if a course has a chain of prerequisite courses upto n levels, all those n courses would be displayed

5. Modify a student's enrollment state (i.e. add/remove a student to/from a class)
   - Includes a sub-menu to allow the user choose from both the options this feature supports

6. List the students enrolled in a particular class

4.3 Backend Implementation

The Backend Implementation in Java is an extremely structured program, with specific standards set and used throughout the app. Various Design Patterns (like Factory Pattern, Singleton Pattern) too were used.
The Backend code uses Oracle's implementation of the JDBC Drivers (or OJDBC Drivers), to communicate with the Oracle Database.
Before booting up, the app expects the absolute path of a config file, containing at least these 3 paramaters – db_url, username, password. These parameters are used to make a connection to the database. So the syntax to start the program is:

```
java –jar SRS.jar /path/to/config/file.ini
```

4.4 Database and PL SQL

The Database is queried by the application using various functions and procedures defined in the package named 'srs'. The list of procedures and functions included in this package is mentioned above (Refer Section 2).
Inside the project directory exists a directory called 'sql', which includes files containing the various PL SQL components, ex. Package spec, body, triggers etc. In order to execute and create these PL SQL definitions on the Database, we just need to execute the file run.sql in the SQLPlus console (file exists in the sql directory).

```
SQL> start run
```