**Documentation**

**CS 532** DATABASE SYSTEMS PROJECT 2

Team Members

1. Arpit Yadav

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. SagarChaudhari

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Shrey Jain

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The Student Registration System uses Oracle’s PL/SQL and JDBC connection and Servlet JSP for web interface. It is a menu driven web application. Below are the stored procedures

1) show\_students

2) show\_tas

3) show\_courses

4) show\_classes

5) show\_enrollment

6) show\_prerequsites

7) show\_logs

8) class\_ta

9) get\_prereq

10) enroll\_student

11) drop\_student

12) delete\_student

We have created package dbproject and all stored procedure are part of same package.

We have created sequence to generate unique number as a primary key in logSeq.sql for logs table

We have also created triggers for below activities in triggers.sql -

* Delete Students.
* Insert tuple in logs table when student is deleted.
* Insert tuple in logs table when student dropped from courses.
* Insert tuple in logs table when student is enrolled.

**If validation fails in any procedure then we are showing the error message on UI**

**1. PROCEDURE show\_students(sys\_cur OUT SYS\_REFCURSOR);**

**2. PROCEDURE show\_tas(sys\_cur OUT SYS\_REFCURSOR);**

**3. PROCEDURE show\_courses(sys\_cur OUT SYS\_REFCURSOR);**

**4. PROCEDURE show\_classes(sys\_cur OUT SYS\_REFCURSOR);**

**5. PROCEDURE show\_enrollments(sys\_cur OUT SYS\_REFCURSOR);**

**6. PROCEDURE show\_prerequisites(sys\_cur OUT SYS\_REFCURSOR);**

**7 PROCEDURE show\_logs(sys\_cur OUT SYS\_REFCURSOR);**

For above seven procedures we have used “SYS\_REFCURSOR” to return the output to UI and these procedures just return all records of respective tables.

**8. PROCEDURE class\_ta(class\_id IN VARCHAR2, ta\_cur OUT SYS\_REFCURSOR, err OUT number);**

This procedure returns the TA details of given classid.

I have done all the validations and used “SYS\_REFCURSOR” to return the output and we have used “err” number variable to return the validation errors to java code.

**9. PROCEDURE get\_prereq(deptcode IN varchar2,course\_# IN number, preResult OUT preReqArray, err OUT number);**

This procedure returns the prerequisites of given course. We have used custom array type “**preReqArray**” to store the output and number variable “err” to return the validation errors.

Custome type: create or replace type preReqArray IS VARRAY(100) OF VARCHAR2(20);

**10. PROCEDURE enroll\_student(b\_# IN VARCHAR2,class\_id IN VARCHAR2,err OUT number);**

This procedure enrolls the new student into a given class and for this procedure we have done all the validations including prerequisites check. We have used “err” number variable to return the validation errors to java code.

If student gets registered successfully then we are showing success message on UI.

**11. PROCEDURE drop\_student(b\_# IN VARCHAR2,class\_id IN VARCHAR2,err OUT number);**

This procedure removes student from class and we have used “err” number variable to return the validation errors to java code.

**12. PROCEDURE delete\_student(b\_# IN VARCHAR2,err OUT number);**

This procedure deletes the student from system.

**Trigers:**

1. **drop\_student:**

This trigger gets invoked after record deletion from enrollments table and decreases the class size by 1(vacancy will be increased by 1) of the class from which student is dropped.

1. **on\_enrollment:**

This trigger gets invoked after record insertion on enrollments table and increases the class size by 1 (vacancy will be decreased by 1) of the class in which student is enrolled.

1. **on\_delete\_std:**

This trigger gets invoked after deletion of record from students table and it deletes all the entries for corresponding student from enrollments table.

1. **log\_enrollment:**

This trigger gets invoked after record gets inserted into enrollments table and it inserts the entry into logs table with all the details including key value as “B#||classid” from enrollments record.

1. **log\_student\_del:**

This trigger gets invoked after the record deletion from students table and it inserts a record into logs table with all the details including key value as “B#” of deleted student.

1. **log\_delete:**

This trigger gets invoked after record gets deleted from enrollments table and it inserts the entry into logs table with all the details including key value as “B#||classid” from enrollments record.

**Work division -**

Sagar -

* Question 4 (Prerequisite Procedure) -

Show\_prerequisite (Direct & Indirect)

Satisfying prerequisite conditions

* Question 5- (Procedure for Enrolling)

Insert tuples into enrollment

Satisfy conditions for enrollment

Rejecting enrollment conditions

Checking prerequisite conditions

Incrementing class size

Implementing Triggers

* Question 6–(Drop Students)

Deleting Students from enrollment

Satisfy all conditions for delete

Prerequisite Check

Triggers Implemented

* Interface

Java code and JDBC Database Connection handling and also web application implementation using Servlet and front-end interaction using JSP.

GUI – Implementing interactive and menu driven interface

* Documentation -

PL/SQL package, stored procedures, triggers, sequence commenting.

Shrey -

* Tables Preparation
* Question 2 (show procedures) -

Show\_students

Show\_Tas

Show\_courses

Show\_classes

Show\_enrollments

Show\_prerequisites

Show\_logs

* Question 8 (Triggers) –

Implementing Triggers

Triggers whenever students are deleted

Triggers for student’s enrollment

Adding tuples to log

* Documentation –

Report and PL/SQL package, triggers, sequence commenting.

* Testing –

Running demo test cases.

Checking outputs

Arpit –

* Question 1 (Create Sequence)-

Log\_seq

Start from 100

Increment by 1

Generating unique values

* Question 3 (Procedure for displaying TA)

Displaying TA information

Conditions to Display

* Question 7 (Procedure to Delete students)

Implemented Triggers

Satisfy all conditions to delete

* Documentation -

PL/SQL package, triggers, sequence commenting.

* Interface

Front End – GUIImplementing interactive and menu driven interface

**Team Meetings**

**Meeting – 1**

**Date –** 11/25/2018

Discussion on Work Division.

Notes review

**Meeting – 2**

**Date –**11/26/2018

Installing Database Locally and PL/SQL Queries

**Meeting – 3**

**Date –** 11/27/2018

PL/SQL Queries

Interface

**Meeting – 4**

**Date –** 11/28/2018

Integration of PL/SQL code with Java code.

**Meeting – 5**

**Date –** 11/29/2018

Final Code Testing and Documentation.

**Experience Gained -**

Sagar -

1) Learned PL/SQL thoroughly including custom types, stored procedures and triggers, cursors.

2) Got good exposure of JDBC connection

3) JSP & Servlet.

Shrey -

1) Experience on writing PL/SQL code.

2) Learned to handle triggers.

3) Building Interface and Connecting.

4) Got good exposure of Testing and Debugging

Arpit –

1) Learned PL/SQL.

2) Got good exposure of implementing Triggers.

3)Learned frond end development