**Project 2 – Using PL/SQL and JDBC to Implement Student Registration System**

**Main Documentation on Procedures/ Triggers/ Sequence.**

**Log Sequence:**

Created a log sequence where the key will start with 1000 and on entry of every next value it will be incremented by 1. When ever a new row is inserted in the log table, this sequence will be used by populating the log# automatically.

**Packages**

Created a package called **student\_reg\_system** with in and out cursor as **student\_cursor.** In the package header all the procedures that are written in the package body are defined here with all the parameters.

In the package body each procedure defined in the package head is written by meeting the requirements given in the project-2.

**Procedures**

1. Created a procedure to display all students table (display\_all\_students):

A new procedure is created where on calling the display\_all\_students procedure, the complete student table will be displayed with the tuples in the student table.

1. Created a procedure to display all courses table (display\_all\_courses):

A new procedure is created where on calling the display\_all\_courses procedure, the complete courses table will be displayed with the tuples in the courses table.

1. Created a procedure to display all course\_credits table (display\_all\_course\_credits):

A new procedure is created where on calling the display\_all\_course\_credits procedure, the complete course\_credits table will be displayed with the tuples in the course\_credits table.

1. Created a procedure to display all classes table (display\_all\_classes):

A new procedure is created where on calling the display\_all\_classes procedure, the complete classes table will be displayed with the tuples in the classes table.

1. Created a procedure to display all g\_enrollments table (display\_all\_g\_enrollments):

A new procedure is created where on calling the display\_all\_g\_enrollments procedure, the complete g\_enrollments table will be displayed with the tuples in the g\_enrollments table.

1. Created a procedure to display all score\_grade table (display\_all\_score\_grade):

A new procedure is created where on calling the display\_all\_score\_grade procedure, the complete score\_grade table will be displayed with the tuples in the score\_grade table.

1. Created a procedure to display all prerequisites table (display\_all\_prerequisites):

A new procedure is created where on calling the display\_all\_prerequisites procedure, the complete prerequisites table will be displayed with the tuples in the prerequisites table.

1. Created a procedure to display all logs table (display\_all\_logs):

A new procedure is created where on calling the display\_all\_prerequisites procedure, the complete logs table will be displayed with the tuples in the logs table.

1. Created a procedure to display the B#, first\_name, last\_name of every student (get\_stud\_info\_by\_class).

A new procedure is created when calling the get\_stud\_info\_by\_class procedure, all the tuples with B#, first\_name, last\_name will be displayed from the students table when classid is given as the param. Also, there are edge cases that are taken of when the classid that is provided is not valid (if it doesn’t exist in class table), then error message will be displayed saying “The classid is invalid”.

1. Created a procedure to display all the prerequisite courses by concatenating the dept\_code, course# (both direct and indirect prerequisites are displayed - get\_prereq\_info\_by\_dept\_course).

A new procedure is created, when calling the get\_prereq\_info\_by\_dept\_course procedure, all the prerequisite courses will be displayed when dept\_code and course# are given as parameters. Initially the given course# and dept\_code from courses will be validating by checking if both these parameters are available in the courses table or not, if the given parameters are not available then, 'The dept\_code || course# does not exist.' message will be displayed. If the params are present then by concatenating them and checking the direct and indirect prerequisite courses the output will be displayed (concatenation of the courses, example – CS532).

1. Created a procedure to enroll a graduate student into a class when the B# and classid are provided as parameters (enroll\_graduate\_students).

A new procedure is created where we can enroll a graduate student into a class by considering various cases whether the given B# and classid are valid or not. We need to consider various cases such as

* 1. The given b# and classid are valid or not.
  2. The given student is a graduate student or not – this can be checked if the given b# is present in g\_enrollment table.
  3. Also, we can enroll the student if the classes year and semester should be of the current semester and year (i.e., 2021 and Spring).
  4. Also, if the class is full, we cannot enroll the student and the student should meet with the requirements with prerequisites with atleast C grade and currently he cannot be enrolled in more than 5 classes.

For all the above cases, the enrollment should be rejected and in all other cases a new tuple should be inserted in the respective g\_enrollments table, class table and the class size should also be incremented.

1. Created a procedure to drop a graduate student from the given class when student b# and classid are given as params (drop\_graduate\_student).

A new procedure is created where we can drop a graduate student by considering various cases such as:

* 1. The given b# and classid are valid or not.
  2. If the given b#, is not present in the g\_enrollments table then the given student is not a graduate student.
  3. We also need to consider the case if the student (by given b#) is present in the given class or not.
  4. Also, the class should be of current semester (2021 and Spring) then only we can drop the student.
  5. Also, if the current class is the last class in the current semester, then also we cannot drop the student.

For all the above cases, the student cannot be dropped, and in remaining cases the student can be dropped from a class, also we need to consider various cases when the student is dropped then the class size should be decreased these can be done using triggers.

1. Created a procedure to delete a student from students table when B# is given as the parameter (delete\_curr\_student).

A new procedure is created where the student’s entry can be deleted from student table as well as g\_enrollment table if some cases are verified, such as:

* 1. The given b# is valid or not.
  2. If the b# is present in g\_enrollments table or not.

For all the above cases, if are checked correctly and considering all the other cases the the student can be deleted from the students table and if the g\_b# present in the g\_enrollements then the entry should also be deleted.

**Triggers:**

1. Trigger - insert\_student\_logs

Whenever an entry (after) inserted in students table, then a new row will be logged into the logs table. Using this trigger automatic insertion will be done.

1. insert\_grad\_enrollment\_logs

Whenever an entry (after) is inserted in the g\_enrollments table, then a new row will be automatically inserted into the logs table using this trigger.

1. delete\_student\_logs

Whenever an entry is deleted from students (after) table, then a new row will be automatically inserted into the logs table using this trigger.

1. delete\_grad\_enrollments\_logs

Whenever an entry is deleted from g\_enrollments (after) table, then a new row will be automatically inserted into the logs table using this trigger.

1. student\_enroll\_update\_class

Whenever an entry is inserted into g\_enrollments (after) table, then a new row will be automatically inserted into the logs table using this trigger. Also, the class size will be updated automatically.

1. drop\_student\_update\_class

Whenever an entry is deleted from the g\_enrollemts (after), then the class\_size will be decreased from the classes table automatically.

1. drop\_stud\_enrollment

Whenever an entry is deleted from students table (before), then the corresponding entry from the g\_enrollments table will be deleted automatically.

**Interface:**

We have developed an interface that is a text-based menu driven interface where we have main options and sub options for selecting and executing the program based on the options that we have selected.

Main Options:

1. Display existing table Data.

**Sub Options**

* 1. Students Table
  2. Courses Table
  3. Prerequisites Table
  4. Classes Table
  5. Enrollments (G\_Enrollments) Table
  6. Logs Table
  7. Score\_Grade Table
  8. Course\_Credits Table

1. Get student details for a given Class.
2. Course prerequisites.
3. Enrol a graduate student.
4. Drop a graduate student enrolment.
5. Delete a student

Inline comments are written in the program itself for coding explanations.