**STUDENT REGISTRATION SYTEM**

DESIGN DOCUMENT

Submitted by

Sai Prudhvi Chode

Raviteja Dupuguntla

**MainClass.Java**

This java file is used for establishing connection to the database using JDBC and used as an interface program which utilizes all the procedures discussed below.It will regenerate the messages that are printed by the dbms\_output package.

We used a switch case to invoke different procedures based on the given input.

We made use of CallableStatement which is the interface used to execute stored procedures in SQL.

**Ex**:- begin pack\_display.show\_students(?); end;

We set the parameter using the CallableStatement object to a OracleTypes.CURSOR and then we execute the stored procedure.

A ResultSet object maintains a cursor pointing to its current row of data.We used it in a while loop to iterate through the result set.

We used a PreparedStatement object to store a SQL statement that is precompiled.This object can be used to execute that statement multiple times efficiently.

**Procedure.sql**

**We created a package which contains 13 procedures,4 triggers and a sequence for generating log values.**

**Sequence will start with 1000 and increment by 1.**

**4 Triggers are implemented to trigger events upon insert/delete on Students/Enrollments table.Triggers are also used to update the logs table accordingly.**

**Out of 13, 6 procedures are used for displaying data in each of 6 tables given in the project specification.**

**Remaining procedures were used to invoke operations like :**

**7) Adding a student to Students table**

**8) Showing classes taken by a particular student**

**9) Showing prerequisites for a particular course**

**10)Listing all the students who have taken or taking a particular class**

**11) Enrolling a student to a particular class**

**12) Dropping a student from a particular class**

**13) Deleting a student from the students table.**

**Procedures(1-6) : Displaying tuples from tables**

**For displaying tuples, we created a procedure for each table that will use a REF CURSOR out parameter. We store the result returned by the REF CURSOR in the ResultSet .We can access data using the ResultSet Object.**

**Example : rs.getString(1)**

**rs -> ResultSet object ,**

**1 -> Pointing to column 1 of result**

**Procedure 7 : Adding a student to Students table**

**For adding a student we created a procedure that takes six input parameters for each attribute of Students table(SID,Firstname,Lastname,Status,GPA,Email) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “a new tuple will be added to the Students table.**

**It will also invoke triggers upon successfully adding a student which will update the logs table accordingly.**

**Procedure 8: Showing classes taken by a particular student**

**For listing the classes taken by a particular student we created a procedure that takes single input parameter (sid) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “all the classes taken by the student with input SID”**

**For displaying the result we used a REF CURSOR out parameter which will return all the columns in the result. We store the result in the ResultSet and access the results using the ResultSet object.**

**The procedure also handles exception cases like Invalid SID and Invalid Classid.**

**Procedure 9: Showing prerequisites for a particular course**

**For displaying the pre-requisites for a particular course, we created a procedure that takes 2 input parameters (dept\_code,course\_no) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “pre-requisites for a particular course are returned”**

**For displaying the result we used a REF CURSOR out parameter which will return all the columns in the result. We store the result in the ResultSet and access the results using the ResultSet object.**

**Example : rs.getString(1) - > contains dept\_code**

**rs.getInt(2) -> contains course\_no**

**Procedure 10 : Listing all the students who have taken or taking a particular class**

**For listing all the students who have taken a particular class we created a procedure that takes single input parameter (classid) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “all the students who has taken or taking the class”**

**For displaying the result we used a REF CURSOR out parameter which will return all the columns in the result. We store the result in the ResultSet and access the results using the ResultSet object.**

**The procedure also handles exception cases like Invalid SID and Invalid Classid.**

**Procedure 11 : Enrolling a student to a particular class**

**For enrolling a student to a particular class we created a procedure that takes 2 input parameters (sid,classid) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “a new tuple will be added to the Enrollments table if all the conditions are satisfied”.**

**The procedure also handles exception cases like Invalid SID and Invalid Classid,Class is closed,Prerequisites not met for the class,Cant enroll in more than 3 classes in the same semester.**

**It will also check the size of class which the student wants to enroll in and display meaningful information.(Ex : If the class\_size = limit it will display “Class is closed”.**

**It will also invoke triggers upon successfully enrolling a student which will update the logs table accordingly.**

**Procedure 12 : Dropping a student from a particular class**

**For dropping a student from a particular class we created a procedure that takes 2 input parameters (sid,classid) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “a tuple will be deleted from the Enrollments table if all the conditions are satisfied”.**

**It will also check the number of classes the student has enrolled in and display meaningful information.(Ex : If the student is dropped and has no classes taken by him it will display “Student is no enrolled in any classes”**

**The procedure also handles exception cases like Invalid SID and Invalid Classid ,Prerequisites Violation,**

**It will also invoke triggers upon successfully dropping a student which will update the logs table accordingly.**

**Procedure 13 : Deleting a student from Students table**

**For deleting a student we created a procedure that takes one input parameters (sid) which will be provided to the procedure by the user input.**

**After we get the required data we execute the procedure and the corresponding operation will be performed which in this case “a tuple will be deleted from the Students table as well as Enrollments table”.**

**The procedure also handles exception cases like Invalid SID.**

**It will also invoke triggers upon successfully deleting a student which will update the logs table accordingly.**