```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

 /*     CS532 - Project 2
 *      1. Shriram Suryawanshi
 *      2. Vinen Furtado
 */

SET SERVEROUTPUT ON;

CREATE OR REPLACE PACKAGE SQLPackage AS

        TYPE myCursor IS REF CURSOR;

-- @@ : Students Module
        PROCEDURE display_students(oCursor OUT myCursor);
        PROCEDURE add_student(temp_sid IN students.sid%TYPE, temp_firstname IN students.firstname%TYPE, temp_lastname IN
        students.lastname%TYPE, temp_status IN students.status%TYPE, temp_gpa IN students.gpa%TYPE, temp_email IN students.email%TYPE);
        PROCEDURE find_student(oCursor IN OUT myCursor,  temp_sid IN students.sid%TYPE);
        PROCEDURE delete_student(temp_sid IN students.sid%TYPE);

-- @@ : Courses Module
        PROCEDURE display_courses(oCursor OUT myCursor);
        PROCEDURE display_prerequisites(oCursor OUT myCursor);
        PROCEDURE find_course(oCursor IN OUT myCursor, temp_dept IN courses.dept_code%TYPE, temp_course IN courses.course_no%TYPE);

-- @@ : Classes Module
        PROCEDURE display_classes(oCursor OUT myCursor);
        PROCEDURE find_class(oCursor IN OUT myCursor,  temp_classid IN classes.classid%TYPE);

-- @@ : Enrollments Module
        PROCEDURE display_enrollments(oCursor OUT myCursor);
        PROCEDURE enroll_student(temp_sid IN students.sid%TYPE, temp_cid IN classes.classid%TYPE);
        PROCEDURE drop_enrollment(temp_sid IN students.sid%TYPE, temp_cid IN classes.classid%TYPE);

-- @@ : logs Module
        PROCEDURE display_logs(oCursor OUT myCursor);

END;
/
```

```
CREATE OR REPLACE PACKAGE BODY SQLPackage AS

-- @@ : Students module
    -- @@ : Students module - show students
    PROCEDURE display_students(oCursor OUT myCursor) AS
    BEGIN
        OPEN oCursor FOR SELECT * FROM students;
    END;

    -- @@ : Students module - insert student
    PROCEDURE add_student(temp_sid IN students.sid%TYPE, temp_firstname IN students.firstname%TYPE, temp_lastname IN
    students.lastname%TYPE, temp_status IN students.status%TYPE, temp_gpa IN students.gpa%TYPE, temp_email IN students.email%TYPE) AS
    BEGIN
        INSERT INTO students VALUES (temp_sid, temp_firstname, temp_lastname, temp_status, temp_gpa, temp_email);
        COMMIT;
    END;

    -- @@ : Students module - find student and his classes
    PROCEDURE find_student(oCursor IN OUT myCursor,  temp_sid IN students.sid%TYPE) AS
        sidcheck varchar(10);
        cnt varchar(10);

    BEGIN
        sidcheck := 0;

        -- @@ : check if sid is present, else throw error and return
        BEGIN
            SELECT sid INTO sidcheck FROM students WHERE sid = temp_sid;
            EXCEPTION
                WHEN no_data_found THEN raise_application_error(-20001, 'The sid is invalid.');
                    RETURN;
        END;

        -- @@ : check if student is enrolled in any class, if yes, skip this block, if no, return only student details
        BEGIN
            SELECT sid INTO cnt FROM enrollments WHERE sid = temp_sid AND ROWNUM = 1;
            EXCEPTION
                WHEN no_data_found THEN
                    OPEN oCursor FOR SELECT sid, lastname, status FROM students WHERE sid = temp_sid;
                    RETURN;
        END;
```

```
        -- @@ : sid is valid, student is enrolled in class, return the student + enrolled class details
        BEGIN
            OPEN oCursor FOR
                SELECT students.sid, students.lastname, students.status, classes.classid, concat(classes.dept_code, classes.course_no),
                courses.title, classes.year, classes.semester FROM Students
                    JOIN enrollments ON enrollments.sid = students.sid AND students.sid = temp_sid
                    JOIN classes ON classes.classid = enrollments.classid
                    JOIN courses ON courses.dept_code = classes.dept_code AND courses.course_no = classes.course_no;
        END;
    END;

    -- @@ : Students module - delete student
    PROCEDURE delete_student(temp_sid IN students.sid%TYPE) AS
        sidcheck varchar(10);

    BEGIN
        sidcheck := 0;

        -- @@ : check if sid is present in DB,  else throw error and return
        BEGIN
            SELECT sid INTO sidcheck FROM students WHERE sid = temp_sid;
            EXCEPTION
                WHEN no_data_found THEN raise_application_error(-20001, 'The sid is invalid.');
                RETURN;
        END;

        -- @@ : delete student if sid is found in db
        BEGIN
            DELETE FROM students WHERE sid = temp_sid;
            COMMIT;
        END;
    END;



-- @@ : Courses module
    -- @@ : Courses module - show courses
    PROCEDURE display_courses(oCursor OUT myCursor) AS
    BEGIN
        OPEN oCursor FOR SELECT * FROM courses;
    END;

    -- @@ : Courses module - show prerequisites
```

```
    PROCEDURE display_prerequisites(oCursor OUT myCursor) AS
    BEGIN
        OPEN oCursor FOR SELECT * FROM prerequisites;
    END;

    -- @@ : Courses module - find course
    PROCEDURE find_course(oCursor IN OUT myCursor, temp_dept IN courses.dept_code%TYPE, temp_course IN courses.course_no%TYPE) AS
        cidcheck varchar(20);
    BEGIN
        cidcheck := 0;

        -- @@ : check if course is available in DB, else throw error and return
        BEGIN
            SELECT title INTO cidcheck FROM courses WHERE course_no = temp_course AND UPPER(dept_code) = temp_dept;
            EXCEPTION
                WHEN no_data_found THEN raise_application_error(-20001, 'Course not found!');
                RETURN;
        END;

        -- @@ : course present : return course and it's prerequisite details
        BEGIN
            OPEN oCursor FOR WITH Parent (pre_dept_code, pre_course_no, dept_code, course_no) AS (
                SELECT pre_dept_code, pre_course_no, dept_code, course_no FROM prerequisites m WHERE UPPER(dept_code) = temp_dept
                AND course_no = temp_course
                UNION ALL
                SELECT  m.pre_dept_code, m.pre_course_no, m.dept_code, m.course_no FROM prerequisites m INNER JOIN Parent p ON
                p.pre_dept_code = m.dept_code and p.pre_course_no = m.course_no
            )
            SELECT concat(pre_dept_code, pre_course_no) FROM Parent;
        END;
    END;


-- @@ : Classes module
    -- @@ : Classes module - show classes
    PROCEDURE display_classes(oCursor OUT myCursor) AS
    BEGIN
        OPEN oCursor FOR SELECT * FROM classes;
    END;

    -- @@ : Classes module - find class and students
    PROCEDURE find_class(oCursor IN OUT myCursor,  temp_classid IN classes.classid%TYPE) AS
        classidcheck char(10);
```

```
        stdntchk char(10);
    BEGIN
        classidcheck := 0;

        -- @@ : check if class id is available in DB, else throw error and return
        BEGIN
            SELECT classid INTO classidcheck FROM classes WHERE classid = temp_classid;
            EXCEPTION
                WHEN no_data_found THEN raise_application_error(-20001, 'The cid is invalid.');
                RETURN;
        END;

        -- @@ : class id correct, but no student enrolled, return the class details only
        BEGIN
            SELECT sid INTO stdntchk FROM enrollments WHERE classid = temp_classid AND ROWNUM = 1;
            EXCEPTION
                WHEN no_data_found THEN
                    OPEN oCursor FOR
                        SELECT classes.classid, courses.title, classes.semester, classes.year FROM classes
                            JOIN courses ON courses.dept_code = classes.dept_code AND courses.course_no = classes.course_no AND
                            classes.classid = temp_classid;
                RETURN;
        END;

        -- @@ : class id is correct, also class has students enrolled, return class + enrolled student details
        BEGIN
            OPEN oCursor FOR
                SELECT classes.classid, courses.title, classes.semester, classes.year, students.sid, students.lastname FROM classes
                    JOIN courses ON courses.dept_code = classes.dept_code AND courses.course_no = classes.course_no AND classes.classid =
                    temp_classid
                    JOIN enrollments ON enrollments.classid = classes.classid
                    JOIN students ON students.sid = enrollments.sid;
        END;
    END;




-- @@ : Enrollments module
    -- @@ : Enrollments module - show enrollments
    PROCEDURE display_enrollments(oCursor OUT myCursor) AS
    BEGIN
        OPEN oCursor FOR SELECT * FROM enrollments;
    END;
```

```
-- @@ : Enrollments module - enroll student
PROCEDURE enroll_student(temp_sid IN students.sid%TYPE, temp_cid IN classes.classid%TYPE) AS
        sidcheck varchar(10);
        classidcheck char(10);
        classcnt number;
        classlimit number;
        enrollcnt number;
        totalenrolled number;
        precourses number;
        temp_dept prerequisites.dept_code%TYPE;
        temp_course prerequisites.course_no%TYPE;

BEGIN
        sidcheck := 0;
        classidcheck := 0;
        classcnt := 0;
        classlimit := 0;
        enrollcnt := 0;
        totalenrolled := 0;
        precourses := 0;

        -- @@ : check is sid is valid, else throw error and return
        BEGIN
                SELECT sid INTO sidcheck FROM students WHERE sid = temp_sid;
                EXCEPTION
                        WHEN no_data_found THEN raise_application_error(-20001, 'The sid is invalid.');
                        RETURN;
        END;

        -- @@ : check is classid is valid, else throw error and return
        BEGIN
                SELECT classid INTO classidcheck FROM classes WHERE classid = temp_cid;
                EXCEPTION
                        WHEN no_data_found THEN raise_application_error(-20001, 'The classid is invalid.');
                        RETURN;
        END;

        -- @@ : checking if classsize = limit, if yes, then throw error and return
        BEGIN
                BEGIN
                        SELECT class_size INTO classcnt FROM classes WHERE classid = temp_cid;
                        SELECT classes.limit INTO classlimit FROM classes WHERE classid = temp_cid;
```

```
        END;
        IF(classcnt = classlimit)
            THEN raise_application_error(-20001, 'The class is closed.');
            RETURN;
        END IF;
END;

-- @@ : check if student is already enrolled in this class, if yes, throw error and return
BEGIN
        BEGIN
            SELECT COUNT(*) INTO enrollcnt FROM enrollments WHERE classid = temp_cid AND sid = temp_sid;
            EXCEPTION WHEN no_data_found THEN enrollcnt := null;
        END;

        IF(enrollcnt = 1)
            THEN raise_application_error(-20001, 'The student is already in the class.');
            RETURN;
        END IF;
END;

-- @@ : check if student has enrolled in two classes for this semester of this year, if yes, then enroll student in new requested classs, and
throw error and return for overloaded
--@@ :  check if student has enrolled in three classes for this semester of this year, if yes then throw error and return
BEGIN
        BEGIN
            SELECT COUNT(temp.classid) AS COUNT INTO totalenrolled FROM
                (SELECT en.sid, en.classid FROM enrollments en JOIN classes cl ON cl.classid = en.classid WHERE en.sid = temp_sid AND
                cl.semester IN  (SELECT semester FROM classes WHERE classid = temp_cid)
                    AND cl.year IN (SELECT classes.year FROM classes WHERE classid = temp_cid)) temp
            GROUP BY temp.sid;
            EXCEPTION WHEN no_data_found THEN totalenrolled := 0;
        END;
        IF(totalenrolled = 2)
            THEN
                BEGIN
                    INSERT INTO enrollments VALUES (temp_sid, temp_cid, null);
                    COMMIT;
                END;
                raise_application_error(-20001, 'You are overloaded.');
            RETURN;
        END IF;

        IF(totalenrolled = 3)
```

```
            THEN raise_application_error(-20001, 'Students cannot be enrolled in more than three classes in the same semester.');
            RETURN;
        END IF;
    END;

    -- @@ : check if the student has completed all the prerequisites with grade C, else throw error and return
    BEGIN
        SELECT dept_code, course_no INTO temp_dept, temp_course FROM classes WHERE classid = temp_cid;

        BEGIN
            SELECT COUNT(*) into precourses FROM (SELECT UNIQUE temp2.dept_code, temp2.course_no, temp3.sid, temp3.lgrade FROM
                (SELECT * FROM (WITH Parent (pre_dept_code, pre_course_no, dept_code, course_no) AS
                    (SELECT pre_dept_code, pre_course_no, dept_code, course_no FROM prerequisites pre WHERE dept_code =
                    temp_dept AND course_no = temp_course
                        UNION ALL
                        SELECT  pre.pre_dept_code, pre.pre_course_no, pre.dept_code, pre.course_no FROM prerequisites pre
                        INNER JOIN Parent prnt ON prnt.pre_dept_code = pre.dept_code AND prnt.pre_course_no = pre.course_no)
                    SELECT pre_dept_code, pre_course_no FROM Parent) temp1 JOIN classes cl ON cl.dept_code =
                    temp1.pre_dept_code AND cl.course_no = temp1.pre_course_no) temp2
                LEFT JOIN
                (SELECT en.sid, en.lgrade, cl.dept_code, cl.course_no FROM enrollments en JOIN classes cl ON cl.classid = en.classid WHERE
                en.sid = temp_sid) temp3
                    ON temp2.dept_code = temp3.dept_code AND temp2.course_no = temp3.course_no) WHERE sid IS NULL OR lgrade >
                    'D';
        END;

        IF(precourses > 0)
            THEN raise_application_error(-20001, 'Prerequisite courses have not been completed.');
            RETURN;
        END IF;
    END;

    -- @@ : all above conditions fail, then enroll student
    BEGIN
        INSERT INTO enrollments VALUES (temp_sid, temp_cid, null);
        COMMIT;
    END;
END;

-- @@ : Enrollments module - drop enrollment
PROCEDURE drop_enrollment(temp_sid IN students.sid%TYPE, temp_cid IN classes.classid%TYPE) AS
    sidcheck varchar(10);
    classidcheck char(10);
```

```
    enrollcnt number;
    precourses number;
    temp_dept prerequisites.dept_code%TYPE;
    temp_course prerequisites.course_no%TYPE;
    remainclass number;
    remainstud number;

BEGIN
    sidcheck := 0;
    classidcheck := 0;
    enrollcnt := 0;
    precourses := 0;
    remainclass := 0;
    remainstud := 0;

    -- @@ ; check if sid is valid, else throw error and return
    BEGIN
        SELECT sid INTO sidcheck FROM students WHERE sid = temp_sid;
        EXCEPTION
            WHEN no_data_found THEN raise_application_error(-20001, 'The sid is invalid.');
            RETURN;
    END;

    -- @@ check is classid is valid, else throw error and return
    BEGIN
        SELECT classid INTO classidcheck FROM classes WHERE classid = temp_cid;
        EXCEPTION
            WHEN no_data_found THEN raise_application_error(-20001, 'The classid is invalid.');
            RETURN;
    END;

    -- @@ : check if student is enrolled in class, if not throw error and return
    BEGIN
        BEGIN
            SELECT COUNT(*) INTO enrollcnt FROM enrollments WHERE classid = temp_cid AND sid = temp_sid;
            EXCEPTION WHEN no_data_found THEN enrollcnt := null;
        END;

        IF(enrollcnt = 0)
            THEN raise_application_error(-20001, 'The student is not enrolled in the class.');
            RETURN;
        END IF;
    END;
```

```
-- @@ : check if this class is acting as prerequisite for any other class, if yes, then throw error and return
BEGIN
    SELECT dept_code, course_no INTO temp_dept, temp_course FROM classes WHERE classid = temp_cid;

    BEGIN
        SELECT COUNT(*) INTO precourses  FROM (SELECT UNIQUE * FROM ( SELECT cl.dept_code, cl.course_no FROM (WITH Child
        (pre_dept_code, pre_course_no, dept_code, course_no) AS
            (SELECT pre_dept_code, pre_course_no, dept_code, course_no FROM prerequisites pre WHERE pre_dept_code =
            temp_dept AND pre_course_no = temp_course
                UNION ALL
                SELECT  pre.pre_dept_code, pre.pre_course_no, pre.dept_code, pre.course_no FROM prerequisites pre
                    INNER JOIN Child chld ON chld.dept_code = pre.pre_dept_code AND chld.course_no = pre.pre_course_no)
            SELECT dept_code, course_no FROM Child) temp1 JOIN classes cl ON cl.dept_code = temp1.dept_code AND cl.course_no =
            temp1.course_no) temp2
                JOIN (SELECT en.sid, cl.course_no, cl.dept_code FROM enrollments en JOIN classes cl ON  en.classid = cl.classid
                WHERE en.sid = temp_sid AND en.classid != temp_cid) temp3
                ON temp2.dept_code = temp3.dept_code AND temp2.course_no = temp3.course_no);
        EXCEPTION WHEN no_data_found THEN precourses := 0;
    END;

    IF(precourses > 0)
        THEN raise_application_error(-20001, 'The drop is not permitted because another class uses it as a prerequisite.');
        RETURN;
    END IF;
END;

-- @@ : none of the above condition matched, then delete student from enrollment
-- @@ : check remaining enrolled classes for this student, if none, throw error - no class and return
-- @@ : check how many student still enrolled for this class, if none, throw error - no students and return
-- @@ : check how many student still enrolled for this class, if none, throw error - no students and return
BEGIN
    DELETE FROM enrollments WHERE classid = temp_cid AND sid = temp_sid;
    COMMIT;

    BEGIN
        SELECT classid INTO remainclass FROM enrollments WHERE sid = temp_sid AND ROWNUM = 1;
        EXCEPTION
            WHEN no_data_found THEN raise_application_error(-20001, 'This student is not enrolled in any classes.');
            RETURN;
    END;

    BEGIN
```

```
                    SELECT class_size INTO remainstud FROM classes WHERE classid = temp_cid AND ROWNUM = 1;
                    EXCEPTION
                        WHEN no_data_found THEN raise_application_error(-20001, 'The class now has no students.');
                        RETURN;
                END;
            END;
        END;

-- @@ : Logs Module
        PROCEDURE display_logs(oCursor OUT myCursor) AS
        BEGIN
            OPEN oCursor FOR SELECT * FROM logs;
        END;

END;
/
```