1. Add a procedure to the school\_api package called remove\_student. This procedure accepts anstudent\_id and returns nothing. Based on the student ID passed in, it removes the student from the database. If the student does not exist or if a problem occurs while removing the student (such as a foreign key constraint violation), let the calling program handle it.

SOLUTION:

/\*package specification\*/

CREATE OR REPLACE PACKAGE school\_api AS

PROCEDURE remove\_student(p\_student\_id IN student.student\_id%TYPE);

END school\_api;

CREATE SEQUENCE INSTRUCTOR\_ID\_SEQ increment by 10;

/\*package body\*/

CREATE OR REPLACE PACKAGE BODY school\_api AS

PROCEDURE remove\_student(p\_student\_id IN student.student\_id%TYPE) IS

BEGIN

DELETE FROM student WHERE student\_id = p\_student\_id;

END;

END school\_api;

SET SERVEROUTPUT ON

DECLARE

v\_student\_id student.student\_id%TYPE := &sv\_student\_id;

BEGIN

school\_api.remove\_student(v\_student\_id);

END;

OUTPUT:

If the id exist in the table:

Error report -

ORA-02292: integrity constraint (SYS.ENR\_STU\_FK) violated - child record found

ORA-06512: at "SYS.SCHOOL\_API", line 4

ORA-06512: at line 4

02292. 00000 - "integrity constraint (%s.%s) violated - child record found"

\*Cause: attempted to delete a parent key value that had a foreign

dependency.

\*Action: delete dependencies first then parent or disable constraint.

Otherwise:

PL/SQL procedure successfully completed.

2. Alter remove\_student in the school\_api package body to accept an additional parameter. This new parameter should be a VARCHAR2 and should be called p\_ri. Make p\_ri default to R. The new parameter may contain a value of R or C. If R is received, it represents DELETE RESTRICT, and the procedure acts as it does now. If there are enrollments for the student, the delete is disallowed. If a C is received, it represents DELETE CASCADE. This functionally means that the remove\_student procedure locates all records for the student in all the Student Database tables. It

removes them from the database before attempting to remove the student from the student table. Decide how to handle the situation when the user passes in a code other than C or R.

SOLUTION:

/\*package specification\*/

CREATE OR REPLACE PACKAGE school\_api AS

PROCEDURE remove\_student(p\_student\_id IN student.student\_id%TYPE,

p\_ri VARCHAR2 DEFAULT 'C');

END school\_api;

/\*package body\*/

CREATE OR REPLACE PACKAGE BODY school\_api AS

PROCEDURE remove\_student(p\_student\_id IN student.student\_id%TYPE,

p\_ri VARCHAR2 DEFAULT 'C') IS

student\_exists EXCEPTION;

not\_valid\_p\_ri EXCEPTION;

BEGIN

IF p\_ri = 'R' THEN

DECLARE

test\_var CHAR(1);

BEGIN

SELECT NULL

INTO test\_var

FROM enrollment e

WHERE e.student\_id = p\_student\_id AND ROWNUM = 1;

RAISE student\_exists;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DELETE FROM student WHERE student\_id = p\_student\_id;

END;

ELSIF p\_ri = 'C' THEN

DELETE FROM enrollment WHERE student\_id = p\_student\_id;

DELETE FROM grade WHERE student\_id = p\_student\_id;

DELETE FROM student WHERE student\_id = p\_student\_id;

ELSE

RAISE not\_valid\_p\_ri;

END IF;

EXCEPTION

WHEN not\_valid\_p\_ri THEN

DBMS\_OUTPUT.PUT\_LINE('Not a valid p\_ri! Error!');

WHEN student\_exists THEN

DBMS\_OUTPUT.PUT\_LINE('Student exist in other tables! Error!');

END;

END school\_api;

SET SERVEROUTPUT ON

DECLARE

v\_student\_id student.student\_id%TYPE := &sv\_student\_id;

BEGIN

school\_api.remove\_student(v\_student\_id);

END;

OUTPUT: