Project Phase II

COMP-4150: Adv & Practical Database Systems

Alina Noor – 105018971

Salwa Mohamed - 105089339

Anuhya Chowdary paturi - 110024295

1. **Get details of if student is placed in room.**

DECLARE

s\_id *VARCHAR2*(8);

fname *VARCHAR2*(8);

room\_status *VARCHAR2*(8);

dob DATE;

room\_num *NUMBER*;

passed\_inspection CHAR(1);

placed *BOOLEAN*;

BEGIN

SELECT student\_id, student\_fname, DOB, current\_status INTO s\_id, fname, dob, room\_status FROM Student WHERE student\_id = 12345678;

IF room\_status = 'placed' THEN

placed := TRUE;

END IF;

IF placed THEN

SELECT fr.room\_num, fi.Satisfaction\_cond INTO room\_num, passed\_inspection FROM Leases le, Flats fl, Flat\_rooms fr, Flat\_Inspections fi WHERE s\_id = le.student\_id AND le.flat\_place\_num = fr.place\_num AND fr.flat\_num = fl.flat\_num AND fl.flat\_num = fi.Flat\_num;

dbms\_output.put\_line('Name - '||fname||' DOB - '||dob||' Room number - '||room\_num||' Passed Inspection -'||passed\_inspection);

ELSE

dbms\_output.put\_line(fname||' isnt placed in a room');

END IF;

END;

2. **Check if a student id corresponds to a student.**

ACCEPT x *VARCHAR2*(8) PROMPT 'Enter the student id';

DECLARE

s\_id *VARCHAR2*(8):= &x;

fname student.student\_fname%TYPE;

dob student.DOB%TYPE;

student\_exists *NUMBER*;

BEGIN

SELECT COUNT(\*) INTO student\_exists FROM Student WHERE student\_id = s\_id;

IF student\_exists > 0 THEN

SELECT student\_fname, dob INTO fname, dob FROM Student WHERE student\_id = s\_id;

dbms\_output.put\_line('ID '||s\_id||' belongs to '||fname||' who was born ' ||dob);

ELSE

dbms\_output.put\_line('ID '||s\_id||' doesnt belong to any student');

END IF;

END;

3. **Check if a student is 21 yet.**

DECLARE

s\_id *VARCHAR2*(8):= 11124376;

fname student.student\_fname%TYPE;

dob student.DOB%TYPE;

student\_exists *NUMBER*;

BEGIN

SELECT COUNT(\*) INTO student\_exists FROM Student WHERE student\_id = s\_id;

IF student\_exists > 0 THEN

SELECT student\_fname, dob INTO fname, dob FROM Student WHERE student\_id = s\_id;

dbms\_output.put\_line('ID '||s\_id||' belongs to '||fname||' who was born ' ||dob);

IF dob > DATE '2002-01-01' THEN

dbms\_output.put\_line('This student is not 21 yet.');

ELSE

dbms\_output.put\_line('This student is 21 or older.');

END IF;

ELSE

dbms\_output.put\_line('ID '||s\_id||' doesnt belong to any student');

END IF;

END;

4. **Raise exception if advisor id < 0 and if id does not exist**.

DECLARE

a\_id Staff\_Advisor.advisor\_id%type := &ad\_id;

a\_fname Staff\_Advisor.advisor\_fname%type;

a\_lname Staff\_advisor.advisor\_lname%type;

a\_dptname Staff\_Advisor.dept\_name%type;

-- User defined Exceptions

invalid\_aid EXCEPTION;

BEGIN

IF a\_id <= 0 THEN

RAISE invalid\_aid;

ELSE

SELECT advisor\_fname, advisor\_lname, deptname INTO a\_fname, a\_lname, a\_dptname FROM Staff\_Advisor

WHERE advisor\_id = a\_id;

DBMS\_OUTPUT.PUT\_LINE ('Advisor ID:' || a\_id);

DBMS\_OUTPUT.PUT\_LINE ('Advisor Name:' || a\_fname || ' ' || a\_lname);

DBMS\_OUTPUT.PUT\_LINE ('Advisors Department:' || a\_dptname);

END IF;

EXCEPTION

WHEN invalid\_id THEN

DBMS\_OUTPUT.PUT\_LINE ('Advisor ID must be greater than Zaro.');

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE ('No Advisor with that ID exists.');

WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE ('Error.');

END;

/

5. **Cursor for returning more than one row, returing all rows with advisor id, name and department name**

DECLARE

a\_id Staff\_Advisor.advisor\_id%type;

a\_fname Staff\_Advisor.advisor\_fname%type;

a\_lname Staff\_advisor.advisor\_lname%type;

a\_dptname Staff\_Advisor.dept\_name%type;

-- Declaring Explicit Cursor

CURSOR a\_advisors IS

SELECT advisor\_id, advisor\_fname, advisor\_lname, dept\_name FROM Staff\_Advisor;

BEGIN

-- Opening Cursor

OPEN a\_advisors;

LOOP

-- Fetching: accessing each row values into declared variables.

FETCH a\_advisors INTO a\_id, a\_fname, a\_lname, a\_dptname;

EXIT WHEN a\_advisors%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE (a\_id || ' ' || a\_fname || ' ' || a\_lname || ' ' ||a\_dptname);

END LOOP;

-- close cursor to release allocated memory

CLOSE a\_advisors;

END;

/

6.**This Function would be helpful to find the advisor of a student based on the student Id**

CREATE OR REPLACE FUNCTION s\_name(stuid *NUMBER*)

RETURN *VARCHAR2*

IS

s\_fname student.student\_fname%TYPE;

s\_lname student.student\_lname%TYPE;

aid student.advisor\_id%TYPE;

BEGIN

SELECT student\_fname,student\_lname,advisor\_id INTO s\_fname,s\_lname,aid FROM student WHERE student.student\_id=stuid;

RETURN ('student ID:'||stuid||'---Student First Name:'||s\_fname||'---Student Last Name:'||s\_lname);

END;

Testing:

DECLARE

stuid *NUMBER* := 11124376; -- pick an student ID to test the function

BEGIN

DBMS\_OUTPUT.PUT\_LINE( s\_name(stuid));

END;

7. T**his procedure would be helpful if a new student needs to be added in the database**

CREATE OR REPLACE PROCEDURE student\_details(

stuid IN Student.student\_id%TYPE,

sfname IN Student.student\_fname%TYPE,

slname IN Student.student\_lname%TYPE,

g12num IN Student.grade\_12\_num%TYPE,

st\_num IN Student.street%TYPE,

city IN Student.city%TYPE,

postalcode IN Student.postal\_code%TYPE,

D\_O\_B IN Student.DOB%TYPE,

Gen IN Student.gender%TYPE,

Degree IN Student.deg\_category%TYPE,

s\_nationality IN Student.nationality%TYPE,

s\_needs IN Student.special\_needs %TYPE,

comments IN Student.additional\_comments %TYPE,

status IN Student.current\_status%TYPE,

s\_program IN Student.program%TYPE,

a\_id IN Student.advisor\_id%TYPE

)

IS

Begin

insert into student values(stuid,sfname,slname,g12num,st\_num,city,postalcode,D\_O\_B,Gen,Degree,s\_nationality,s\_needs,comments,status,s\_program,a\_id);

dbms\_output.put\_line('Student table Updated');

end;

**--- Testing:**

BEGIN

student\_details(11524376,'Smith','Will',107,'308 - 15th Ave','Windsor ','N9C1A9E',DATE '2002-08-08','Female','Post Graduate','Canadian','Dietary','pollen Allergy','placed','Computer science',58173019);

END;

8. **This package counts the hostel staff and counts the student population so that its useful to compare the student to staff ratio.**

CREATE   
OR   
replace package staff\_count IS FUNCTION count\_staffRETURN number;FUNCTION count\_studentRETURN number;END  
staff\_count;CREATE   
OR   
replace package body staff\_count IS FUNCTION count\_staffRETURN number IS s\_result number;BEGIN   
 SELECT *Count*(\*)   
 INTO s\_result   
 FROM hostel\_staff ;   
   
 RETURN s\_result;   
END;FUNCTION count\_studentRETURN number IS stu\_result number;BEGIN   
 SELECT *Count*(\*)   
 INTO stu\_result   
 FROM student ;   
   
 RETURN stu\_result;   
END;END   
staff\_count;

**TESTING:**

BEGIN   
 dbms\_output.put\_line ('Number of staff in the hostel residence is :');   
 dbms\_output.put\_line (staff\_count.count\_staff);   
 dbms\_output.put\_line ('Number of Students in the hostel residence is :');   
 dbms\_output.put\_line (staff\_count.count\_student);   
end;

9. **Creating trigger for every insert, update and delete into Staff\_Advisor table.**

CREATE OR REPLACE TRIGGER display\_advisor\_changes

BEFORE DELETE OR INSERT OR UPDATE OF dept\_name

ON Staff\_Advisor

FOR EACH ROW

BEGIN

CASE

WHEN INSERTING THEN

dbms\_output.put\_line('Inserting row into Staff\_Advisor table');

WHEN UPDATING('dept\_name') THEN

dbms\_output.put\_line('Updating Department for a row in Staff\_Advisor');

WHEN DELETING THEN

dbms\_output.put\_line('Deleting a row in Staff\_Advisor table');

END CASE;

END;

/

-- testing INSERT

INSERT INTO Staff\_advisor VALUES('58134919', 'Jane', 'Doe','ResidenceAdvisor ','Residence',2265791236,254);

INSERT INTO Staff\_advisor VALUES('53473019', 'Bob', 'Brent','Advisor ','Hall',2265723876,230);

Select \* from Staff\_Advisor;

-- testing UPDATE

UPDATE Staff\_Advisor

SET dept\_name = 'Flats'

WHERE advisor\_id = '58134919';

Select \* from Staff\_Advisor;

-- testing DELETE

DELETE FROM Staff\_Advisor

WHERE Advisor\_id = '53473019';

Select \* from Staff\_Advisor;