1,

CREATE OR REPLACE PROCEDURE mine(v\_visaexpdate VARCHAR2, v\_type VARCHAR2) IS

v\_day VARCHAR2(20);

v\_count NUMBER(5);

e\_wrong\_type EXCEPTION;

e\_invalid\_format EXCEPTION;

vtype VARCHAR2(20);

BEGIN

IF upper(v\_type) = 'P' THEN

vtype := 'PROCEDURE';

ELSIF upper(v\_type) = 'F' THEN

vtype := 'FUNCTION';

ELSIF upper(v\_type) = 'B' THEN

vtype := 'PACKAGE';

ELSE

RAISE e\_wrong\_type;

END IF;

IF substr(v\_visaexpdate, 1, 2) > 12 OR substr(v\_visaexpdate, 1, 2) < 0 THEN

raise e\_invalid\_format;

END IF;

SELECT to\_char(last\_day(to\_date(v\_visaexpdate,'MM/YY')),'DAY') INTO v\_day FROM dual;

SELECT count(\*) INTO v\_count FROM user\_objects WHERE object\_type= vtype;

dbms\_output.put\_line('Last day of the month '|| v\_visaexpdate || ' is ' || v\_day );

dbms\_output.put\_line('Number of stored objects of type '||v\_type||' is '|| v\_count);

EXCEPTION

WHEN e\_wrong\_type THEN

DBMS\_OUTPUT.PUT\_LINE('You have entered an Invalid letter for the stored object. Try P, F or B.');

WHEN e\_invalid\_format THEN

DBMS\_OUTPUT.PUT\_LINE('You have entered an Invalid FORMAT for the MONTH and YEAR. Try MM/YY.');

END mine;

/

SET serveroutput ON

--EXECUTE mine ('11/09','P')

--EXECUTE mine ('12/09','f')

--EXECUTE mine ('01/10','T')

EXECUTE mine ('13/09','P')

2,

CREATE OR REPLACE PROCEDURE add\_zip (

v\_zip IN zipcode.zip%TYPE,

v\_city IN zipcode.city%TYPE,

v\_state IN zipcode.state%TYPE,

v\_flag OUT VARCHAR2,

v\_currows OUT NUMBER )

IS

vzip VARCHAR2(20);

BEGIN

SELECT count(zip) INTO vzip FROM zipcode WHERE zip = v\_zip;

IF vzip = 0 THEN

INSERT INTO zipcode

VALUES(v\_zip, v\_city, v\_state, 'DBS501\_173A32', to\_char(sysdate, 'DD-MON-YY'), 'DBS501\_173A32', to\_char(sysdate, 'DD-MON-YY'));

v\_flag := 'SUCCESS';

ELSE

DBMS\_OUTPUT.PUT\_LINE('This ZIPCODE '|| v\_zip ||' is already in the Dataase. Try again.');

v\_flag := 'FAILURE';

END IF;

select count(\*) into v\_currows from zipcode WHERE state = v\_state;

END add\_zip;

/

set serveroutput on

--1) test add\_zip(18104, 'Chicago', 'MI', :vflag, :vcount)

VARIABLE vflag VARCHAR2(10)

VARIABLE vcount NUMBER

EXECUTE add\_zip(18104, 'Chicago', 'MI', :vflag, :vcount)

PRINT vflag vcount

SELECT \* FROM zipcode WHERE state = 'MI';

rollback

--2) test add\_zip(48104, 'Chicago', 'MI', :vflag, :vcount)

VARIABLE vflag VARCHAR2(10)

VARIABLE vcount NUMBER

EXECUTE add\_zip(48104, 'Chicago', 'MI', :vflag, :vcount)

PRINT vflag vcount

SELECT \* FROM zipcode WHERE state = 'MI';

3,

CREATE OR REPLACE FUNCTION exist\_zip(v\_zip IN zipcode.zip%TYPE)

RETURN boolean IS

vzip VARCHAR2(20);

BEGIN

SELECT count(zip) INTO vzip FROM zipcode WHERE zip = v\_zip;

IF vzip > 0 THEN

RETURN true;

ELSE

RETURN FALSE;

END IF;

END exist\_zip;

/

CREATE OR REPLACE PROCEDURE add\_zip2 (

v\_zip IN zipcode.zip%TYPE,

v\_city IN zipcode.city%TYPE,

v\_state IN zipcode.state%TYPE,

v\_flag OUT VARCHAR2,

v\_currows OUT NUMBER )

IS

BEGIN

IF exist\_zip(v\_zip) THEN

INSERT INTO zipcode

VALUES(v\_zip, v\_city, v\_state, 'DBS501\_173A32', to\_char(sysdate, 'DD-MON-YY'), 'DBS501\_173A32', to\_char(sysdate, 'DD-MON-YY'));

v\_flag := 'SUCCESS';

ELSE

DBMS\_OUTPUT.PUT\_LINE('This ZIPCODE '|| v\_zip ||' is already in the Dataase. Try again.');

v\_flag := 'FAILURE';

END IF;

SELECT count(\*) INTO v\_currows FROM zipcode WHERE state = v\_state;

END add\_zip2;

/

set serveroutput on

--1) test add\_zip(18104, 'Chicago', 'MI', :vflag, :vcount)

VARIABLE vflag VARCHAR2(10)

VARIABLE vcount NUMBER

EXECUTE add\_zip(18104, 'Chicago', 'MI', :vflag, :vcount)

PRINT vflag vcount

SELECT \* FROM zipcode WHERE state = 'MI';

rollback

--2) test add\_zip(48104, 'Chicago', 'MI', :vflag, :vcount)

VARIABLE vflag VARCHAR2(10)

VARIABLE vcount NUMBER

EXECUTE add\_zip(48104, 'Chicago', 'MI', :vflag, :vcount)

PRINT vflag vcount

SELECT \* FROM zipcode WHERE state = 'MI';

4,

CREATE OR REPLACE FUNCTION instruct\_status (

fname# instructor.first\_name%TYPE,

lname# instructor.last\_name%TYPE

) RETURN NUMBER IS

v\_lname VARCHAR2(10);

v\_countsec NUMBER(3);

e\_no\_schedule EXCEPTION;

BEGIN

SELECT last\_name INTO v\_lname FROM instructor

WHERE UPPER(first\_name)= fname# AND UPPER(last\_name) = lname#;

IF v\_lname IS NULL THEN

RAISE no\_data\_found;

END IF;

SELECT COUNT(\*) INTO v\_countsec FROM section

WHERE instructor\_id = (

SELECT instructor\_id FROM instructor

WHERE UPPER(first\_name)= fname# AND UPPER(last\_name) = lname#

);

IF v\_countsec = 0 THEN

RAISE e\_no\_schedule;

END IF;

RETURN v\_countsec;

EXCEPTION

WHEN no\_data\_found THEN

RETURN -1;

WHEN e\_no\_schedule THEN

RETURN 0;

END instruct\_status;

/

--test 1

DECLARE

cursor c1 is select first\_name, last\_name from instructor order by last\_name;

v\_instructor\_record c1%ROWTYPE;

countsec NUMBER(2);

BEGIN

OPEN c1;

LOOP

FETCH c1 INTO v\_instructor\_record;

EXIT WHEN c1%NOTFOUND;

countsec := instruct\_status(UPPER(v\_instructor\_record.first\_name), UPPER(v\_instructor\_record.last\_name));

IF countsec = 0 THEN

dbms\_output.put\_line(v\_instructor\_record.last\_name||' This Instructor is NOT scheduled to teach.');

ELSIF countsec = -1 THEN

dbms\_output.put\_line('There is NO such instructor');

ELSE

IF countsec > 9 THEN

dbms\_output.put\_line('This Instructor will teach ' || countsec ||' courses and needs a vacation');

ELSE

dbms\_output.put\_line('This Instructor will teach ' || countsec ||' courses');

END IF;

END IF;

END LOOP;

CLOSE c1;

END;

/

--test 2 & test 3

SET SERVEROUTPUT ON

SET VERIFY OFF

ACCEPT fname PROMPT 'Enter first name in upper case'

ACCEPT lname PROMPT 'Enter last name in upper case'

DECLARE

countsec NUMBER(2);

BEGIN

countsec := instruct\_status('&&fname', '&&lname');

IF countsec = 0 THEN

dbms\_output.put\_line('This Instructor is NOT scheduled to teach.');

ELSIF countsec = -1 THEN

dbms\_output.put\_line('There is NO such instructor');

ELSE

IF countsec > 9 THEN

dbms\_output.put\_line('This Instructor will teach ' || countsec ||' courses and needs a vacation');

ELSE

dbms\_output.put\_line('This Instructor will teach ' || countsec ||' courses');

END IF;

END IF;

END;