1. Create an associative array with the element type of a user-defined record. This record should contain the first name, last name, and total number of courses that a particular instructor teaches. Display the records of the associative array on the screen.

SOLUTION:

SET SERVEROUTPUT ON

DECLARE

CURSOR instructor\_cursor IS

SELECT i.first\_name, i.last\_name, COUNT(UNIQUE s.course\_no) t\_courses

FROM instructor i, section s

WHERE i.instructor\_id = s.instructor\_id

GROUP BY i.first\_name, i.last\_name;

TYPE instructor\_record\_type IS RECORD

(first\_name VARCHAR2(25),

last\_name VARCHAR2(25),

total\_courses NUMBER(2) );

TYPE instructor\_type IS TABLE OF instructor\_record\_type

INDEX BY BINARY\_INTEGER;

instructor\_tab instructor\_type;

v\_counter INTEGER := 0;

BEGIN

FOR instructor\_record in instructor\_cursor LOOP

v\_counter := v\_counter + 1;

instructor\_tab(v\_counter).first\_name := instructor\_record.first\_name;

instructor\_tab(v\_counter).last\_name := instructor\_record.last\_name;

instructor\_tab(v\_counter).total\_courses := instructor\_record.t\_courses;

DBMS\_OUTPUT.PUT\_LINE(instructor\_tab(v\_counter).first\_name||' '

||instructor\_tab(v\_counter).last\_name||' '||

'has'||' '||instructor\_tab(v\_counter).total\_courses||' courses!');

END LOOP;

END;

OUTPUT:

Anca Pup has 2 courses!

Alexa Iuga has 3 courses!

Andrei Opra has 2 courses!

2. Modify the script you just created. Instead of using an associative array, use a nested table.

SOLUTION:

SET SERVEROUTPUT ON

DECLARE

CURSOR instructor\_cursor IS

SELECT i.first\_name, i.last\_name, COUNT(UNIQUE s.course\_no) t\_courses

FROM instructor i, section s

WHERE i.instructor\_id = s.instructor\_id

GROUP BY i.first\_name, i.last\_name;

TYPE instructor\_record\_type IS RECORD

(first\_name VARCHAR2(25),

last\_name VARCHAR2(25),

total\_courses NUMBER(2) );

TYPE instructor\_type IS TABLE OF instructor\_record\_type;

instructor\_tab instructor\_type := instructor\_type();

v\_counter INTEGER := 0;

BEGIN

FOR instructor\_record in instructor\_cursor LOOP

v\_counter := v\_counter + 1;

instructor\_tab.EXTEND;

instructor\_tab(v\_counter).first\_name := instructor\_record.first\_name;

instructor\_tab(v\_counter).last\_name := instructor\_record.last\_name;

instructor\_tab(v\_counter).total\_courses := instructor\_record.t\_courses;

DBMS\_OUTPUT.PUT\_LINE(instructor\_tab(v\_counter).first\_name||' '

||instructor\_tab(v\_counter).last\_name||' '||

'has'||' '||instructor\_tab(v\_counter).total\_courses||' courses!');

END LOOP;

END;

OUTPUT:

Anca Pup has 2 courses!

Alexa Iuga has 3 courses!

Andrei Opra has 2 courses!

3. Modify the script you just created. Instead of using a nested table, use a varray.

SOLUTION:

SET SERVEROUTPUT ON

DECLARE

CURSOR instructor\_cursor IS

SELECT i.first\_name, i.last\_name, COUNT(UNIQUE s.course\_no) t\_courses

FROM instructor i, section s

WHERE i.instructor\_id = s.instructor\_id

GROUP BY i.first\_name, i.last\_name;

TYPE instructor\_record\_type IS RECORD

(first\_name VARCHAR2(25),

last\_name VARCHAR2(25),

total\_courses NUMBER(2) );

TYPE instructor\_type IS VARRAY(50) OF instructor\_record\_type;

instructor\_varray instructor\_type := instructor\_type();

v\_counter INTEGER := 0;

BEGIN

FOR instructor\_record in instructor\_cursor LOOP

v\_counter := v\_counter + 1;

instructor\_varray.EXTEND;

instructor\_varray(v\_counter).first\_name := instructor\_record.first\_name;

instructor\_varray(v\_counter).last\_name := instructor\_record.last\_name;

instructor\_varray(v\_counter).total\_courses := instructor\_record.t\_courses;

DBMS\_OUTPUT.PUT\_LINE(instructor\_varray(v\_counter).first\_name||' '

||instructor\_varray(v\_counter).last\_name||' '||

'has'||' '||instructor\_varray(v\_counter).total\_courses||' courses!');

END LOOP;

END;

OUTPUT:

Anca Pup has 2 courses!

Alexa Iuga has 3 courses!

Andrei Opra has 2 courses!

4. Create a user-defined record with four fields: course\_no, description, cost, and prerequisite\_rec. The last field, prerequisite\_rec, should be a user-defined record with three fields: prereq\_no, prereq\_desc, and prereq\_cost. For any ten courses that have a prerequisite course, populate the user-defined record with all the corresponding data, and display its information on the screen.

SOLUTION:

SET SERVEROUTPUT ON

DECLARE

CURSOR course\_cursor IS

SELECT course\_no, description, cost, prerequisite

FROM course

WHERE prerequisite IS NOT NULL AND rownum <= 10;

TYPE prerequisite\_rec\_record\_type IS RECORD

(

prereq\_no course.prerequisite%TYPE,

prereq\_desc course.description%TYPE,

prereq\_cost course.cost%TYPE

);

TYPE course\_record\_type IS RECORD

(course\_no NUMBER(38),

description VARCHAR2(50),

cost NUMBER(9,2),

prerequisite\_rec prerequisite\_rec\_record\_type

);

course\_record course\_record\_type;

BEGIN

FOR course\_rec IN course\_cursor LOOP

course\_record.course\_no := course\_rec.course\_no;

course\_record.description := course\_rec.description;

course\_record.cost := course\_rec.cost;

SELECT course\_no, description, cost

INTO course\_record.prerequisite\_rec.prereq\_no,

course\_record.prerequisite\_rec.prereq\_desc,

course\_record.prerequisite\_rec.prereq\_cost

FROM course

WHERE course\_no = course\_rec.prerequisite;

DBMS\_OUTPUT.PUT\_LINE('Course with ID: '||course\_record.course\_no|| ' is: '

||course\_record.description|| ' costs: '||course\_record.cost||' require subjects: '

||course\_record.prerequisite\_rec.prereq\_no);

END LOOP;

END;

OUTPUT: