CPIT340

HomeWork1 (LAB)

Jeelan Majid Alotaibi - 1907030
Elaf Yousef Aloufi - 1911265

HW1-a

```
DECLARE

/* declare job_sal record type that has two fields:

     1- job_title of the same type as jobs.job_title

     2- job_avg of the same type as employees.salary  */

 type job_sal is record (

        job_title hr.jobs.job_title%type,

        job_avg hr.employees.salary%type    );

/* declare a nested table type nestedRslt where the elements are of the record type job_sal*/

 /* declare a nested table job_avg_sal_nested of type nestedRslt*/

   type nestedRslt is table of job_sal;

   job_avg_sal_nested nestedRslt;
```

```
/* declare an associative array type job_associative_type where the key is job title and the value is average salary

    declare an associative array job_avg_sal of the type job_associative_type*/

        type job_associative_type is table of hr.employees.salary%type index by hr.jobs.job_title%type;

        job_avg_sal job_associative_type;

 /* declare counter1 to loop through the nested table*/

 /* declare counter2 to loop through the associative array*/

   counter1 number := 1;

   counter2 hr.jobs.job_title%type := 1;


BEGIN

/* write a query to retrieve job title and average salary for each job*/

/* use BULK COLLECT INTO to store the query results in the nested table*/

  SELECT hr.jobs.job_title ,avg(E.salary)

  BULK COLLECT INTO job_avg_sal_nested

  FROM hr.employees E natural join hr.jobs

  GROUP BY hr.jobs.job_title;
 /* loop through the nested table and store each item in the assiciative array */
    FOR counter1 IN job_avg_sal_nested.FIRST..job_avg_sal_nested.LAST LOOP
    -- DBMS_OUTPUT.PUT_LINE(job_avg_sal_nested(counter1).job_title ||' : '||job_avg_sal_nested(counter1).job_avg);
    job_avg_sal(job_avg_sal_nested(counter1).job_title) := job_avg_sal_nested(counter1).job_avg;
    END LOOP;
/* loop through the associative array and print the job_title and the average salary for each job*/
    counter2 := job_avg_sal.FIRST;
       DBMS_OUTPUT.PUT_LINE(RPAD('Title', 60, ' ')|| RPAD('|Salary', 30,' '));
       DBMS_OUTPUT.PUT_LINE(RPAD('--', 70, '-'));
    WHILE counter2 IS NOT NULL LOOP
        DBMS_OUTPUT.PUT_LINE(RPAD(TO_CHAR(counter2), 60, ' ') || RPAD(TO_CHAR('|'|| job_avg_sal(counter2)), 30, ' '));
        -- DBMS_OUTPUT.PUT_LINE(TO_CHAR(counter2)|| ' is ' || TO_CHAR(job_avg_sal(counter2)));
        counter2 := job_avg_sal.NEXT(counter2);
    END LOOP;
END;
```

HW1-b

```
DECLARE
    -- PL SQL code to create and fill a two-dimensional array
    -- create VARRAY type of 10 integers
    TYPE array_10_int  IS VARRAY(10) of PLS_INTEGER;
    -- create VARRAY type of array_10_int
    TYPE grid_100_int IS VARRAY(10) of array_10_int;
  -- declare a variable of the grid_100_int type
    grid_var grid_100_int;
    -- declare counters
    i PLS_INTEGER := 0;
    j PLS_INTEGER ;
    numbers PLS_INTEGER;

BEGIN
    -- TO DO : use nested loop to fill grid_var with numbers 1- 100
    /** YOUR CODE HERE **/
    numbers :=0;
  grid_var :=grid_100_int(array_10_int(),array_10_int(),array_10_int(),
              array_10_int(),array_10_int(),array_10_int(),
              array_10_int(),array_10_int(),array_10_int(),array_10_int());
  for i in grid_var.first..grid_var.last loop
    grid_var(i):= array_10_int(null,null,null,null,null,null,null,null,null,null);
    for j in grid_var(i).first..grid_var(i).last loop
     numbers := numbers + 1;
     grid_var(i)(j):= numbers;
    end loop;
  end loop;
-- Print the grid with nested loop
  i:=0;
  LOOP --outer loop
    i := i+1;
    j := 0;
    LOOP    -- inner loop
      j:= j+1;
      IF grid_var(i)(j) < 10 THEN
        DBMS_OUTPUT.PUT(' ' || grid_var(i)(j) || ' ');
      ELSE
        DBMS_OUTPUT.PUT( grid_var(i)(j) || ' ');
      END IF;
      EXIT WHEN (j =10);
    END LOOP;
    dbms_output.put_line(' ');
    EXIT WHEN (i =10);
  END LOOP;
END;
```