Additional Practices

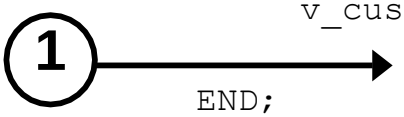
Additional Practices 1 and 2

Note: These exercises can be used for extra practice for declaring variables and writing executable statements.

1. Evaluate each of the following declarations. Determine which of them are not legal and explain why.
   a.
```
DECLARE
   name,dept    VARCHAR2(14);
```
   b.
```
DECLARE
   test         NUMBER(5);
```
   c.
```
DECLARE
   MAXSALARY          NUMBER(7,2) = 5000;
```
   d.
```
DECLARE
   JOINDATE           BOOLEAN := SYSDATE;
```
2. In each of the following assignments, determine the data type of the resulting expression.
   a. `email := firstname || to_char(empno);`
   b. `confirm := to_date('20-JAN-1999', 'DD-MON-YYYY');`
   c. `sal := (1000*12) +  500`
   d. `test := FALSE;`
   e. `temp := temp1 < (temp2/ 3);`
   f. `var := sysdate;`

Additional Practice 3

3.
```
DECLARE
      v_custid   NUMBER(4) := 1600;
      v_custname     VARCHAR2(300) := 'Women Sports Club';
      v_ new_custid  NUMBER(3) := 500;
BEGIN
DECLARE
      v_custid    NUMBER(4) := 0;
      v_custname VARCHAR2(300) := 'Shape up Sports Club';
      v_new_custid NUMBER(3) := 300;
      v_new_custname VARCHAR2(300) := 'Jansports Club';
BEGIN
      v_custid := v_new_custid;
      v_custname := v_custname || ' ' ||  v_new_custname;
```




```
END;
```


```
      v_custid := (v_custid *12) / 10;
END;
      /
```

Evaluate the PL/SQL block given above and determine the data type and value of each of the following variables according to the rules of scoping:

a. The value of `v_custid` at position 1 is:
b. The value of `v_custname` at position 1 is:
c. The value of `v_new_custid` at position 2 is:
d. The value of `v_new_custname` at position 1 is:
e. The value of `v_custid` at position 2 is:
f. The value of `v_custname` at position 2 is:

Additional Practices 4 and 5

Note: These exercises can be used for extra practice when discussing how to interact with the Oracle server and write control structures

4. Write a PL/SQL block to accept a year and check whether it is a leap year. For example, if the year entered is 1990, the output should be “1990 is not a leap year.”
   Hint: The year should be exactly divisible by 4 but not divisible by 100, or it should be divisible by 400.

Additional Practices 4 and 5

Test your solution with the following years:

1990	Not a leap year
2000	Leap year
1996	Leap year
1886	Not a leap year
1992	Leap year
1824	Leap year

```
anonymous block completed
1990 is not a leap year
```

5. a. For the exercises below, you must create a temporary table to store the results. You can either create the table yourself or run the `lab_ap_05.sql` script that will create the table for you. Create a table named `TEMP` with the following three columns:

Column Name	NUM_STORE	CHAR_STORE	DATE_STORE
Key Type			
Nulls/Unique			
FK Table			
FK Column			
Data Type	Number	VARCHAR2	Date
Length	7,2	35	

b. Write a PL/SQL block that contains two variables, `V_MESSAGE` and `V_DATE_WRITTEN`. Declare `V_MESSAGE` as `VARCHAR2` data type with a length of 35 and `V_DATE_WRITTEN` as `DATE` data type. Assign the following values to the variables:

Variable	**Contents**
`V_MESSAGE`	This is my first PL/SQL program
`V_ DATE_WRITTEN`	Current date

Store the values in appropriate columns of the `TEMP` table. Verify your results by querying the `TEMP` table.

NUM_STORE	CHAR_STORE	DATE_STORE
(null)	This is my first PLSQL Program	27-JUN-07

Additional Practices 6 and 7

6. a. Store a department number in a substitution variable.
   b. Write a PL/SQL block to print the number of people working in that department.

```
anonymous block completed
6 employee(s) work for department number 30
```

7. Write a PL/SQL block to declare a variable called `sal` to store the salary of an employee. In the executable part of the program, do the following:
   a. Store an employee name in a substitution variable.
   b. Store his or her salary in the `v_sal` variable.
   c. If the salary is less than 3,000, give the employee a raise of 500 and display the message "<Employee Name>'s salary updated" in the window.
   d. If the salary is more than 3,000, print the employee's salary in the format, "<Employee Name> earns..... "
   e. Test the PL/SQL block for the following last names:

LAST_NAME	SALARY
Pataballa	4800
Greenberg	12000
Ernst	6000

Additional Practice 8 and 9

8. Write a PL/SQL block to store the salary of an employee in a substitution variable.
   In the executable part of the program, do the following:
   - Calculate the annual salary as salary * 12.
   - Calculate the bonus as indicated below:

AnnualSalary	Bonus
>= 20,000	2,000
19,999 - 10,000	1,000
<= 9,999	500

   Display the amount of the bonus in the window in the following format:

   "The bonus is $… ....."

- Test the PL/SQL for the following test cases:

SALARY	BONUS
5000	2000
1000	1000
15000	2000

Note: These exercises can be used for extra practice when discussing how to work with composite data types and cursors and how to handle exceptions.

9. a. Execute the `lab_ap_09_a.sql` script to create a temporary table called `emp`. Write a PL/SQL block to store an employee number, the new department number, and the percentage increase in the salary in substitution variables.

   b. Update the department ID of the employee with the new department number, and update the salary with the new salary. Use the `emp` table for the updates. After the update is complete, display the message "Update complete" in the window. If no matching records are found, display "No Data Found." Test the PL/SQL block for the following test cases:

EMPLOYEE_ID	NEW_DEPARTMENT_ID	% INCREASE	MESSAGE
100	20	2	Update Complete
10	30	5	No Data found
126	40	3	Update Complete

Additional Practices 10 and 11

10. Create a PL/SQL block to declare a cursor `EMP_CUR` to select the employee name, salary, and hire date from the `employees` table. Process each row from the cursor, and if the salary is greater than 15,000 and the hire date is later than 01-FEB-1988, display the employee name, salary, and hire date in the window in the format shown in the sample output below:

```
anonymous block completed
Kochhar earns 17000 and joined the organization on 21-SEP-89
De Haan earns 17000 and joined the organization on 13-JAN-93
```

11. Create a PL/SQL block to retrieve the last name and department ID of each employee from the `EMPLOYEES` table for those employees whose `EMPLOYEE_ID` is less than 114. With the values retrieved from the `employees` table, populate two PL/SQL tables, one to store the records of the employee last names and the other to store the records of their department IDs. Using a loop, retrieve the employee name information and the salary information from the PL/SQL tables and display it in the window, using `DBMS_OUTPUT.PUT_LINE`. Display these details for the first 15 employees in the PL/SQL tables.

```
anonymous block completed
Employee Name: King Department_id: 90
Employee Name: Kochhar Department_id: 90
Employee Name: De Haan Department_id: 90
Employee Name: Hunold Department_id: 60
Employee Name: Ernst Department_id: 60
Employee Name: Austin Department_id: 60
Employee Name: Pataballa Department_id: 60
Employee Name: Lorentz Department_id: 60
Employee Name: Greenberg Department_id: 100
Employee Name: Faviet Department_id: 100
Employee Name: Chen Department_id: 100
Employee Name: Sciarra Department_id: 100
Employee Name: Urman Department_id: 100
Employee Name: Popp Department_id: 100
Employee Name: Raphaely Department_id: 30
```

Additional Practices 12, 13, and 14

12. a. Create a PL/SQL block that declares a cursor called `DATE_CUR`. Pass a parameter of the `DATE` data type to the cursor and print the details of all the employees who have joined after that date.

```
DEFINE B_HIREDATE = 08-MAR-00
```

b. Test the PL/SQL block for the following hire dates: `08-MAR-00`, `25-JUN-97`, `28-SEP-98`, `07-FEB-99`.

```
anonymous block completed
166 Ande 24-MAR-00
167 Banda 21-APR-00
173 Kumar 21-APR-00
```

13. Execute the `lab_ap_09_a.sql` script to re-create the `emp` table. Create a PL/SQL block to promote clerks who earn more than 3,000 to the job title `SR CLERK` and increase their salaries by 10%. Use the `EMP` table for this practice. Verify the results by querying the `emp` table.
Hint: Use a cursor with the `FOR UPDATE` and `CURRENT OF` syntaxes.

14. a. For the exercise below, you will require a table to store the results. You can create the `analysis` table yourself or run the `lab_ap_14_a.sql` script that creates the table for you. Create a table called `analysis` with the following three columns:

Column Name	ENAME	YEARS	SAL
Key Type			
Nulls/Unique			
FK Table			
FK Column			
Data Type	VARCHAR2	Number	Number
Length	20	2	8,2

b. Create a PL/SQL block to populate the `analysis` table with the information from the `employees` table. Use a substitution variable to store an employee's last name.

Additional Practices 12, 13, and 14 (continued)

c. Query the `employees` table to find out whether the number of years that the employee has been with the organization is greater than five; and if the salary is less than 3,500, raise an exception. Handle the exception with an appropriate exception handler that inserts the following values into the `analysis` table: employee last name, number of years of service, and the current salary. Otherwisedisplay `Not due for a raise` in the window. Verify the results by querying the `analysis` table. Use the following test cases to test the PL/SQL block:

LAST_NAME	MESSAGE
Austin	Not due for a raise
Nayer	Due for a raise
Fripp	Not due for a raise
Khoo	Due for a raise