 



Database Programming with PL/SQL 2-4: Using Scalar Data Types

Practice Activities

# Vocabulary

Identify the vocabulary word for each definition below:

|  |  |
| --- | --- |
| BOOLEAN | A datatype that stores one of the three possible values used for logical calculations: TRUE, FALSE, or NULL. |
| Atributul %TYPE | Attribute used to declare a variable according to another previously declared variable or database column. |

# Try It / Solve It

1. Declarations:
   1. Which of the following variable declarations are valid?

|  |  |  |
| --- | --- | --- |
|  | **Declaration** | **Valid or Invalid** |
| a | number\_of\_students PLS\_INTEGER; | INVALID |
| b | STUDENT\_NAME VARCHAR2(10) = Johnson; | INVALID |
| c | stu\_per\_class CONSTANT NUMBER; | INVALID |
| d | tomorrow DATE := SYSDATE+1; | VALID |

* 1. For the invalid declarations above, describe why they are invalid.

La pct.a variabila declarata nu este initializata, la fel si la punctul c. La punctul b sirul de caractere trebuie sa apara intre apostroafe si in loc de = trebuie :=

Write an anonymous block in which you declare and print (on the screen) each of the variables in 1A above, correcting the invalid declarations and adding information as needed.

DECLARE number\_of\_students PLS\_INTEGER := 10;

student\_name VARCHAR2(10) := 'Johnson';

stu\_per\_class CONSTANT NUMBER := 5;

tomorrow DATE := SYSDATE + 1;

BEGIN

DBMS\_OUTPUT.PUT\_LINE(number\_of\_students);

DBMS\_OUTPUT.PUT\_LINE(student\_name);

DBMS\_OUTPUT.PUT\_LINE(stu\_per\_class);

DBMS\_OUTPUT.PUT\_LINE(tomorrow);

END;

1. Evaluate the variables in the following code. Answer the following questions about each variable. Is it named well? Why or why not? If it is not named well, what would be a better name and why?

DECLARE

country\_name VARCHAR2(50); median\_age NUMBER(6, 2);

BEGIN

SELECT country\_name, median\_age INTO country\_name, median\_age FROM countries

WHERE country\_name = 'Japan';

DBMS\_OUTPUT.PUT\_LINE('The median age in '|| country\_name || ' is '

|| median\_age || '.');

END;

In cadrul variabilelor poate sa apara prefixul v\_ , iar in cazul constantelor prefixul c\_ . In acest caz notatiile pentru variabile nu au fost cele mai inspirate intrucat numele sunt identice cu nume de coloane.

Executia acestui program nu va genera eroare intrucat el stie ca in SELECT apar nume de coloane si in INTO nume de variabile

1. Change the declarations in #2 above so they use the %TYPE attribute.

DECLARE

country\_name countries.country\_name %TYPE;

median\_age countries.median\_age %TYPE;

BEGIN

SELECT country\_name, median\_age INTO country\_name, median\_age FROM countries

WHERE country\_name = 'Japan';

DBMS\_OUTPUT.PUT\_LINE('The median age in '|| country\_name || ' is '

|| median\_age || '.');

END;

1. In your own words, describe why using the %TYPE attribute is better than hard-coding data types. Can you explain how you could run into problems in the future by hard-coding the data types of the country\_name and median\_age variables in question 2?

Definirea cu %TYPE este utila atunci cand actualizam baza de date. Spre exemplu daca utilizam tipul de date dintr-o coloana atunci se va actualiza si tipul variabilei care face referire la acea coloana. De asemenea, prin acest mod de definire se evita erorile cauzate de o precizie gresita.Compilatorul determina tipul si size-ul variabilelor cand blocul este compilat

1. Create the following anonymous block:

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Hello World'); END;

* 1. Add a declarative section to this PL/SQL block. In the declarative section, declare the following variables:
* A variable named TODAY of datatype DATE. Initialize TODAY with SYSDATE.
* A variable named TOMORROW with the same datatype as TODAY. Use the %TYPE attribute to declare this variable.
  1. In the executable section, initialize the TOMORROW variable with an expression that calculates tomorrow’s date (add 1 to the value in TODAY). Print the value of TODAY and TOMORROW after printing ‘Hello World’.

DECLARE

TODAY DATE := SYSDATE;

TOMORROW TODAY%TYPE := TODAY + 1;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Hello World');

DBMS\_OUTPUT.PUT\_LINE(today);

DBMS\_OUTPUT.PUT\_LINE(tomorrow);

END;

Copyright © 2020, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.