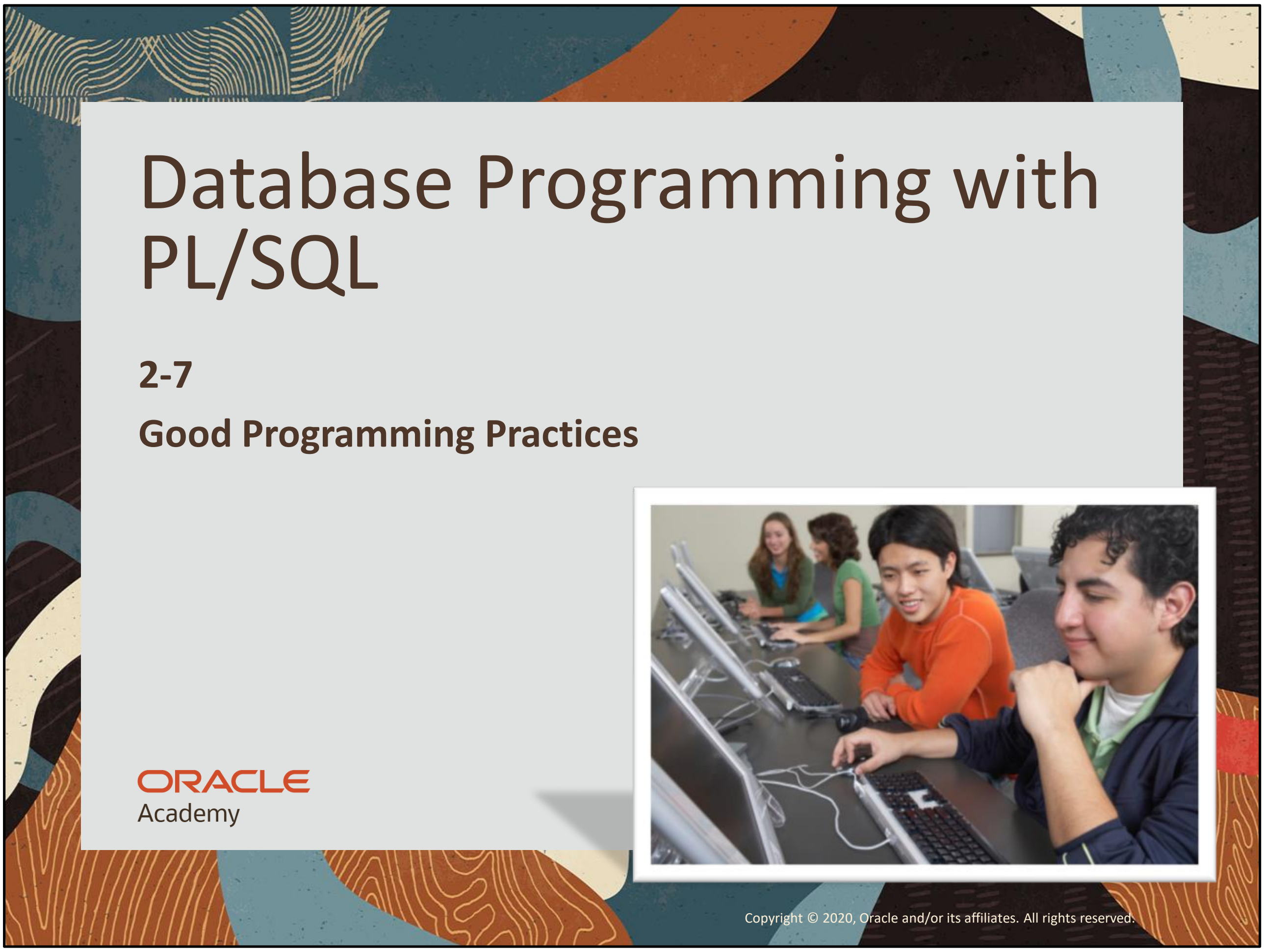
ORACLE
Academy



Database Programming with PL/SQL

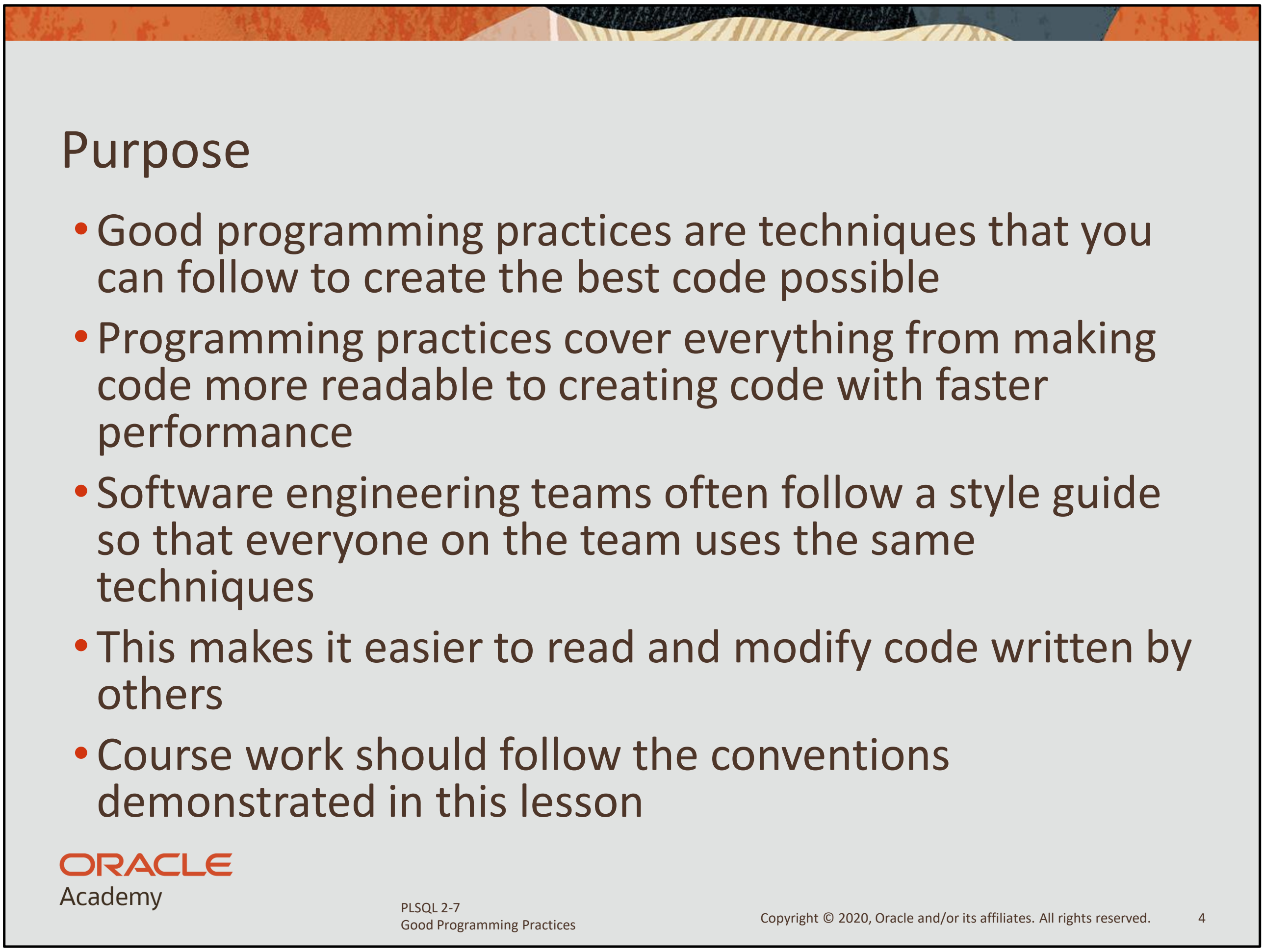
2-7

Good Programming Practices

ORACLE
Academy

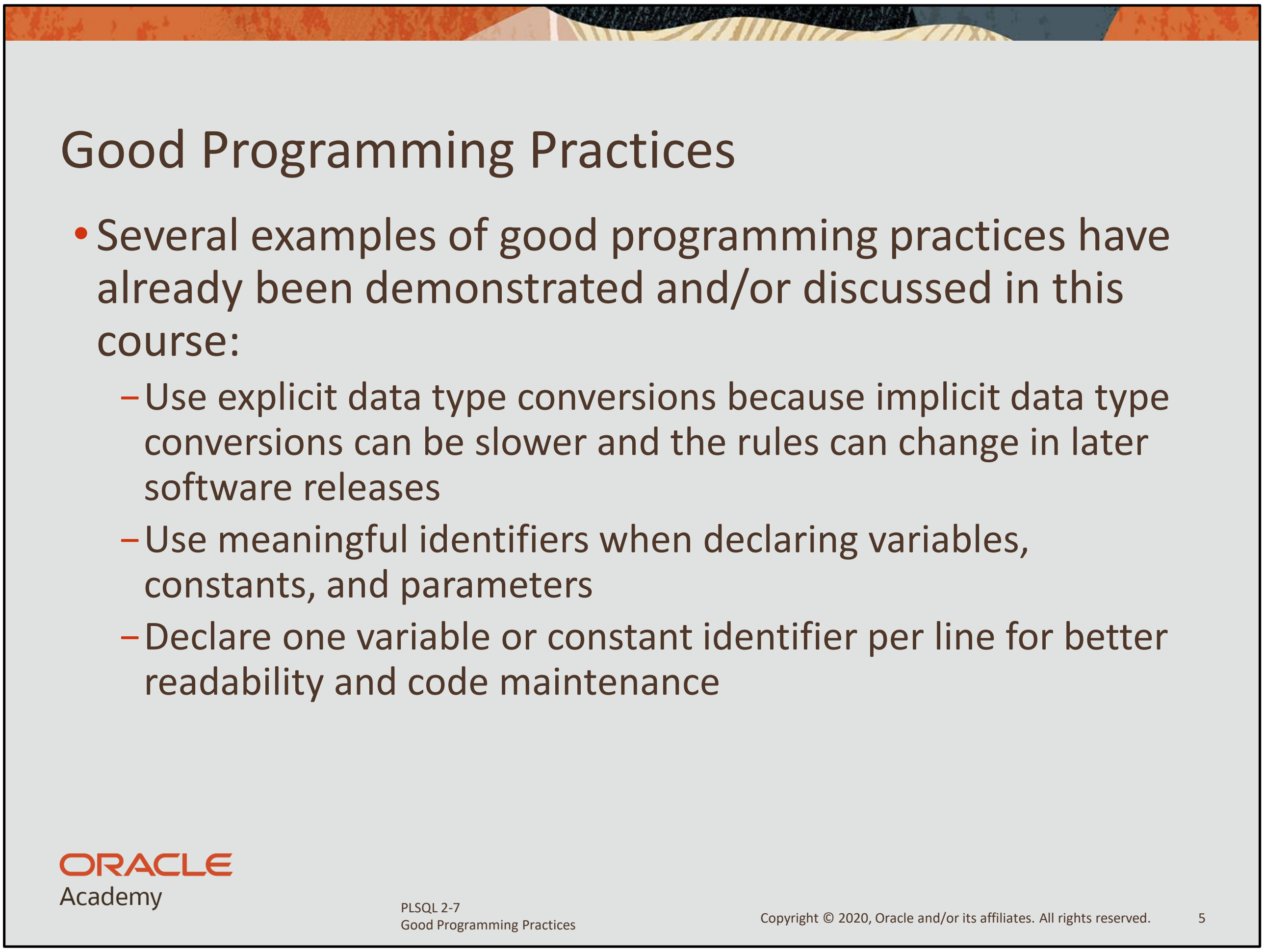
Objectives

- This lesson covers the following objectives:
  - List examples of good programming practices
  - Accurately insert comments into PL/SQL code
  - Create PL/SQL code that follows formatting guidelines to produce readable code



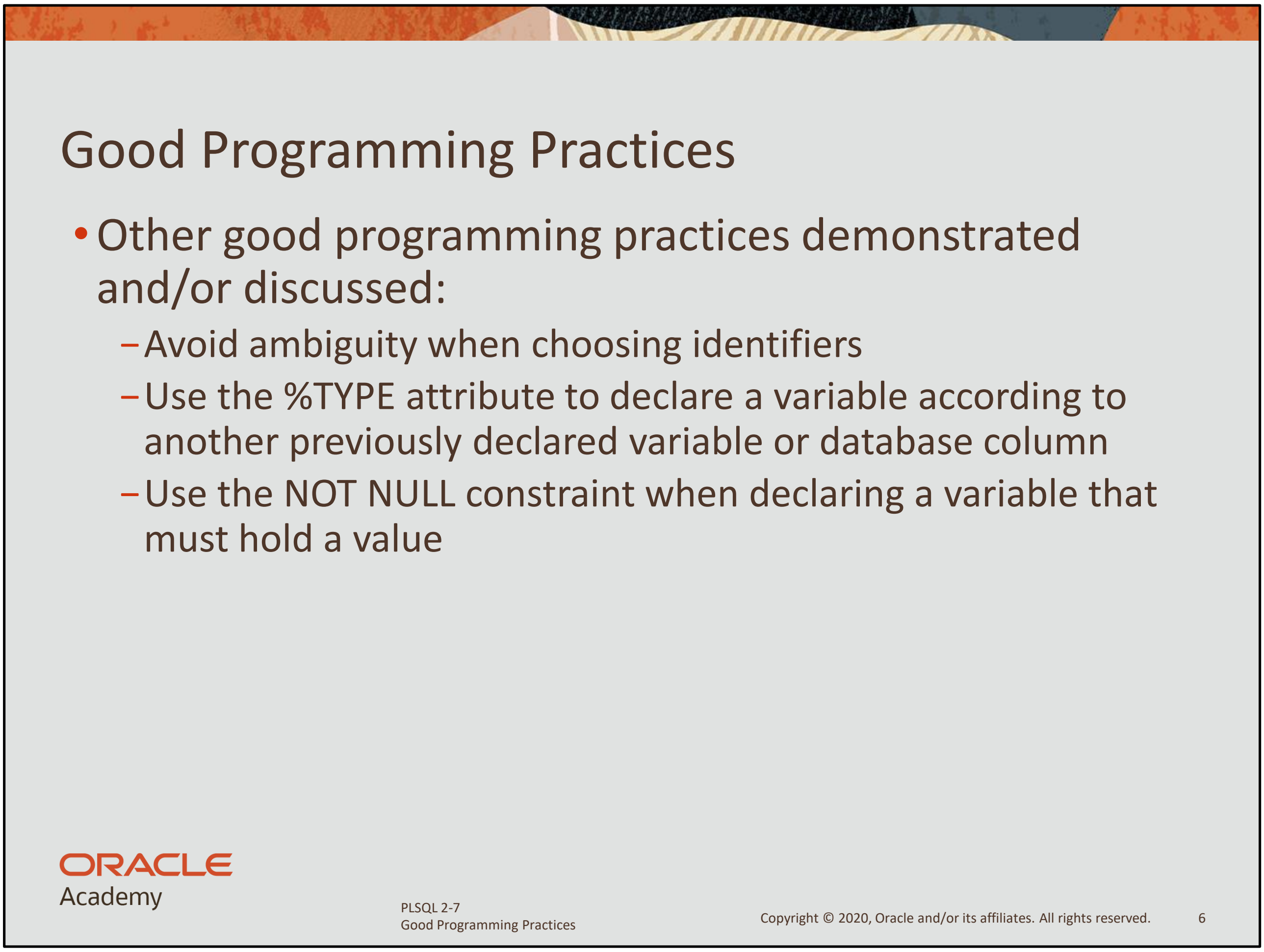
Purpose

- Good programming practices are techniques that you can follow to create the best code possible
- Programming practices cover everything from making code more readable to creating code with faster performance
- Software engineering teams often follow a style guide so that everyone on the team uses the same techniques
- This makes it easier to read and modify code written by others
- Course work should follow the conventions demonstrated in this lesson



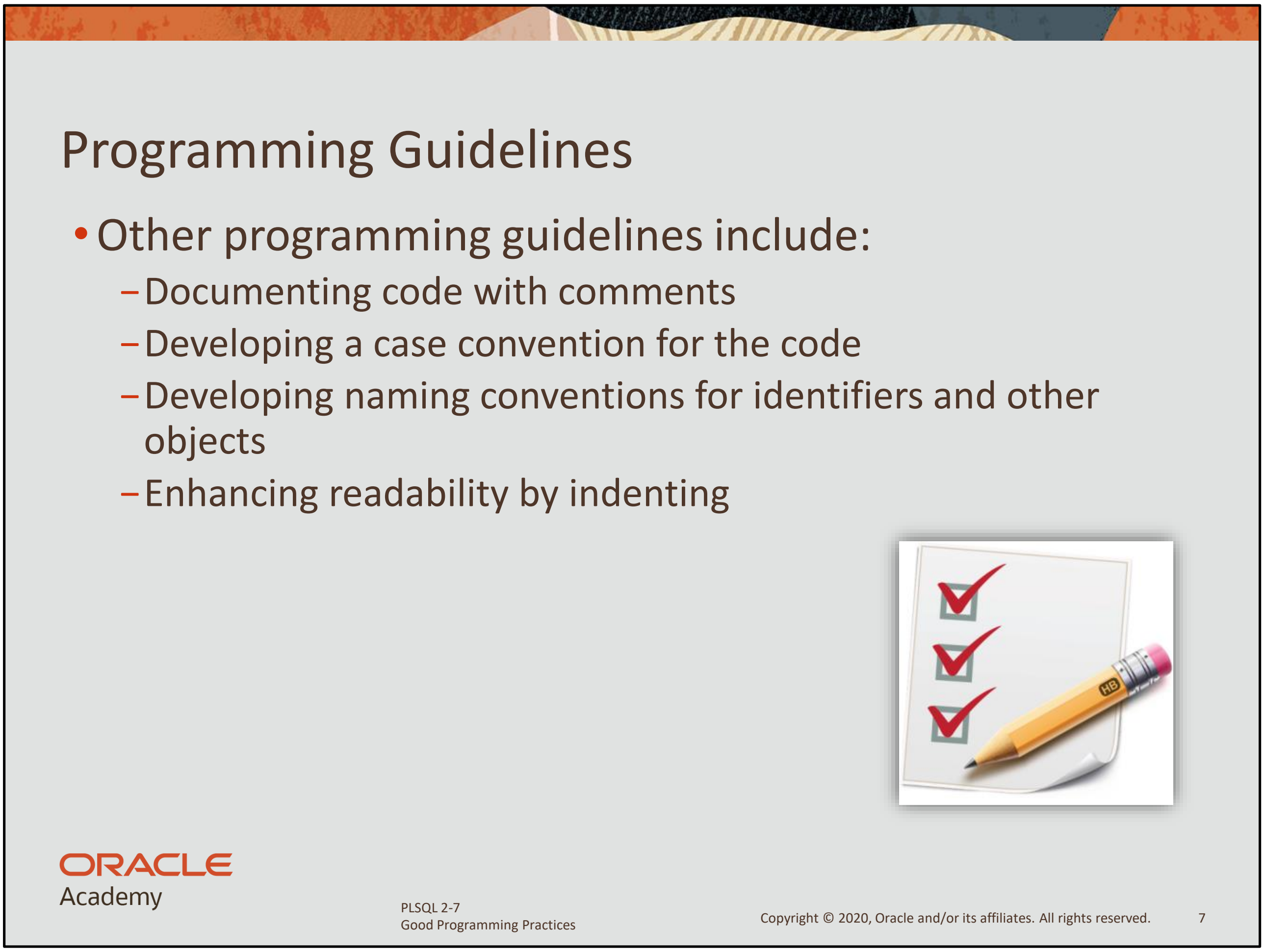
Good Programming Practices

- Several examples of good programming practices have already been demonstrated and/or discussed in this course:
  - Use explicit data type conversions because implicit data type conversions can be slower and the rules can change in later software releases
  - Use meaningful identifiers when declaring variables, constants, and parameters
  - Declare one variable or constant identifier per line for better readability and code maintenance



Good Programming Practices

- Other good programming practices demonstrated and/or discussed:
  - Avoid ambiguity when choosing identifiers
  - Use the %TYPE attribute to declare a variable according to another previously declared variable or database column
  - Use the NOT NULL constraint when declaring a variable that must hold a value



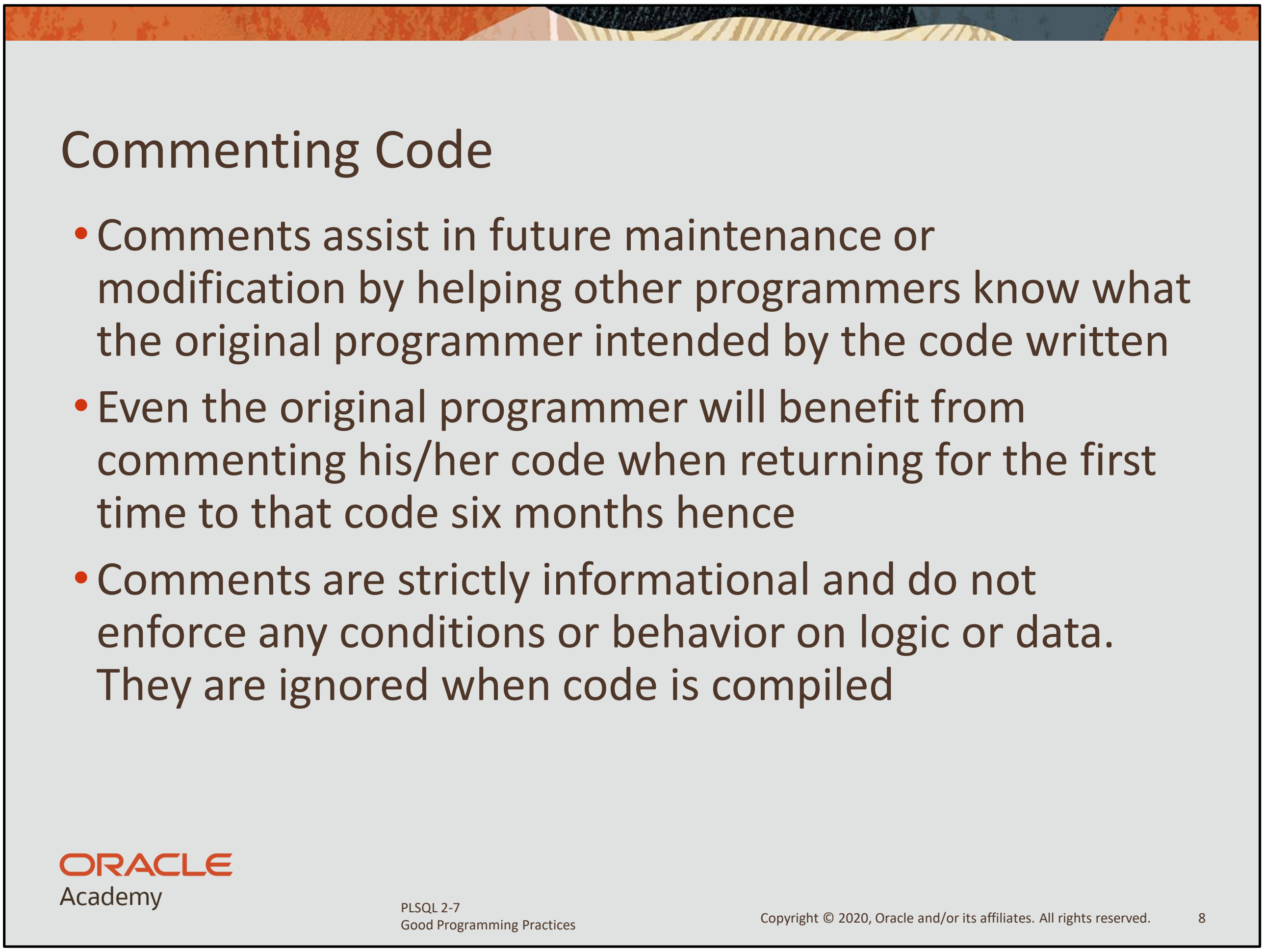
Programming Guidelines

- Other programming guidelines include:
  - Documenting code with comments
  - Developing a case convention for the code
  - Developing naming conventions for identifiers and other objects
  - Enhancing readability by indenting

ORACLE
Academy

Each organization will typically develop and require its own programming guidelines and conventions.

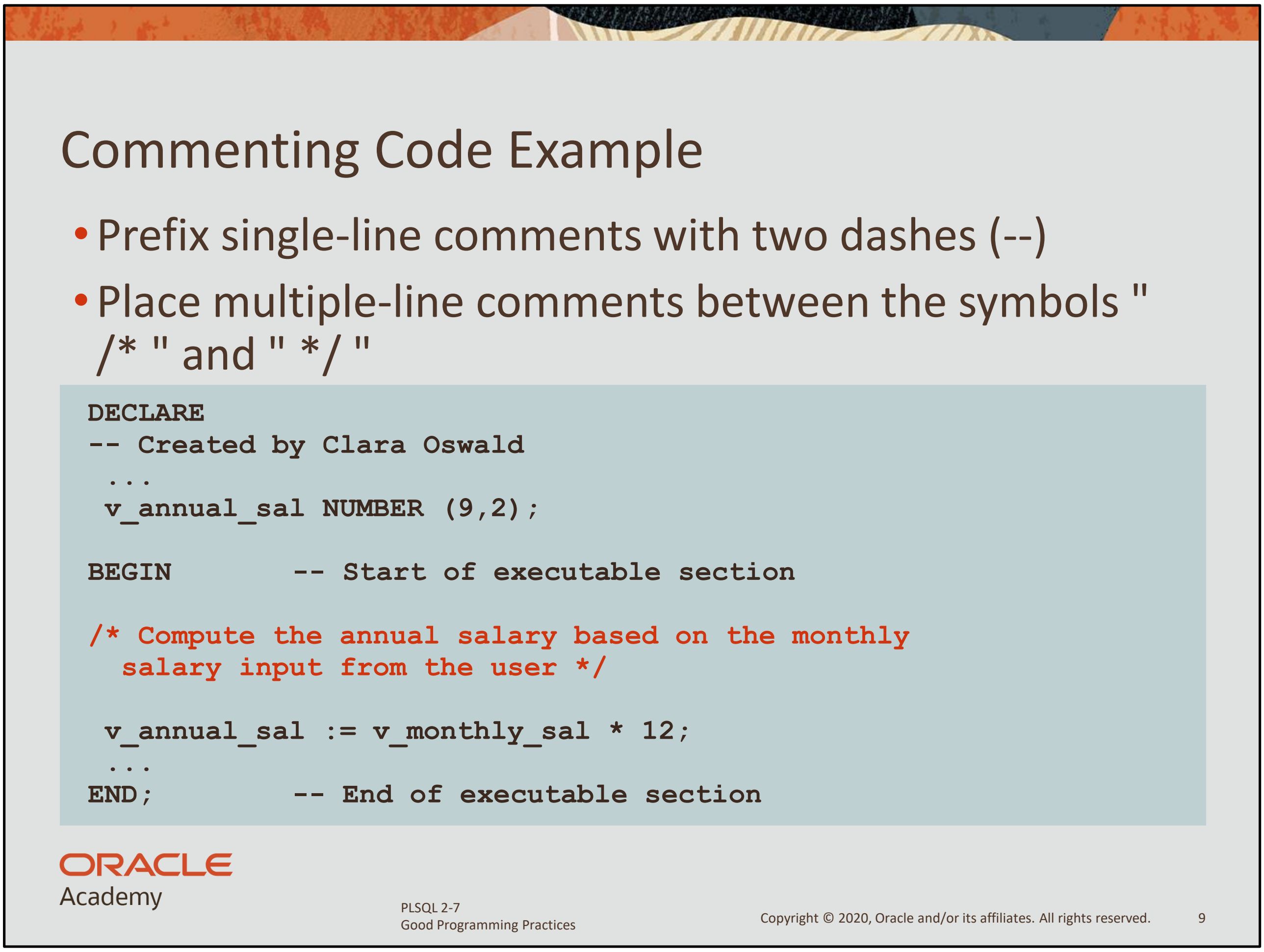
In a sense, it doesn’t matter which conventions are adopted, so long as a meaningful convention exists and is used consistently.



Commenting Code

- Comments assist in future maintenance or modification by helping other programmers know what the original programmer intended by the code written
- Even the original programmer will benefit from commenting his/her code when returning for the first time to that code six months hence
- Comments are strictly informational and do not enforce any conditions or behavior on logic or data. They are ignored when code is compiled

ORACLE
Academy



Commenting Code Example

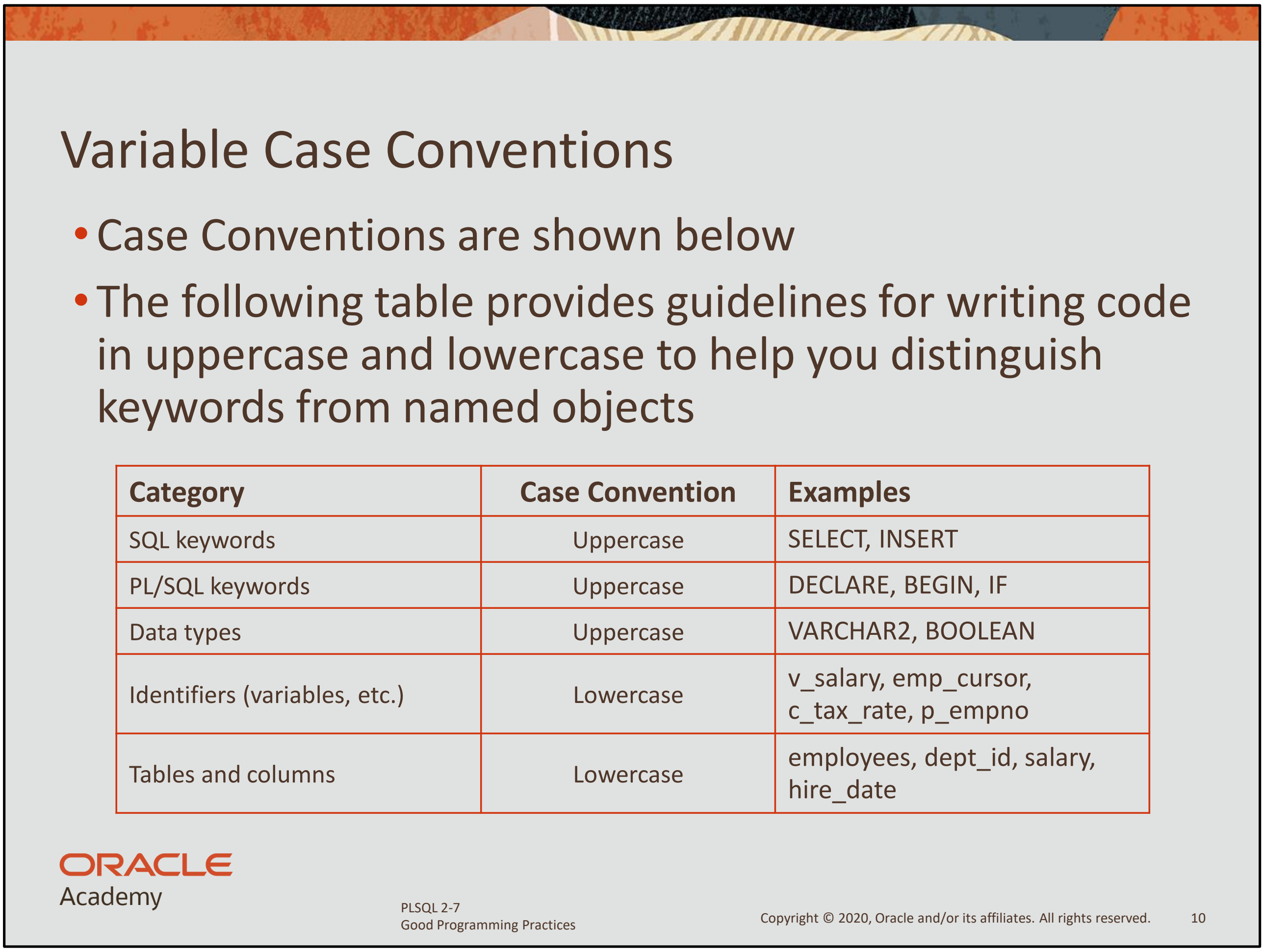
- Prefix single-line comments with two dashes (--)
- Place multiple-line comments between the symbols " /* " and " */ "

```
DECLARE
-- Created by Clara Oswald
 ...
 v_annual_sal NUMBER (9,2);

BEGIN       -- Start of executable section

/* Compute the annual salary based on the monthly
  salary input from the user */

 v_annual_sal := v_monthly_sal * 12;
 ...
END;        -- End of executable section
```



Variable Case Conventions

- Case Conventions are shown below
- The following table provides guidelines for writing code in uppercase and lowercase to help you distinguish keywords from named objects

Category	Case Convention	Examples
SQL keywords	Uppercase	SELECT, INSERT
PL/SQL keywords	Uppercase	DECLARE, BEGIN, IF
Data types	Uppercase	VARCHAR2, BOOLEAN
Identifiers (variables, etc.)	Lowercase	v_salary, emp_cursor, c_tax_rate, p_empno
Tables and columns	Lowercase	employees, dept_id, salary, hire_date

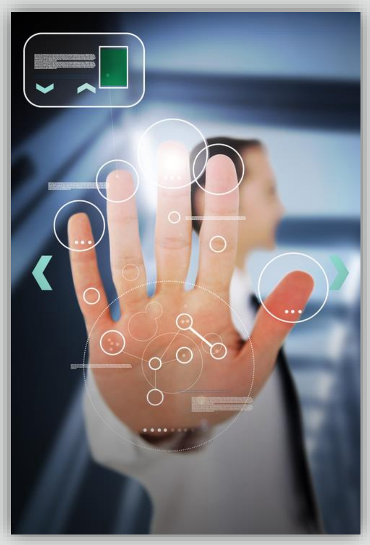
The case convention described here is commonly used in SQL and PL/SQL, and is also the one used in the Oracle product documentation.

Naming Conventions

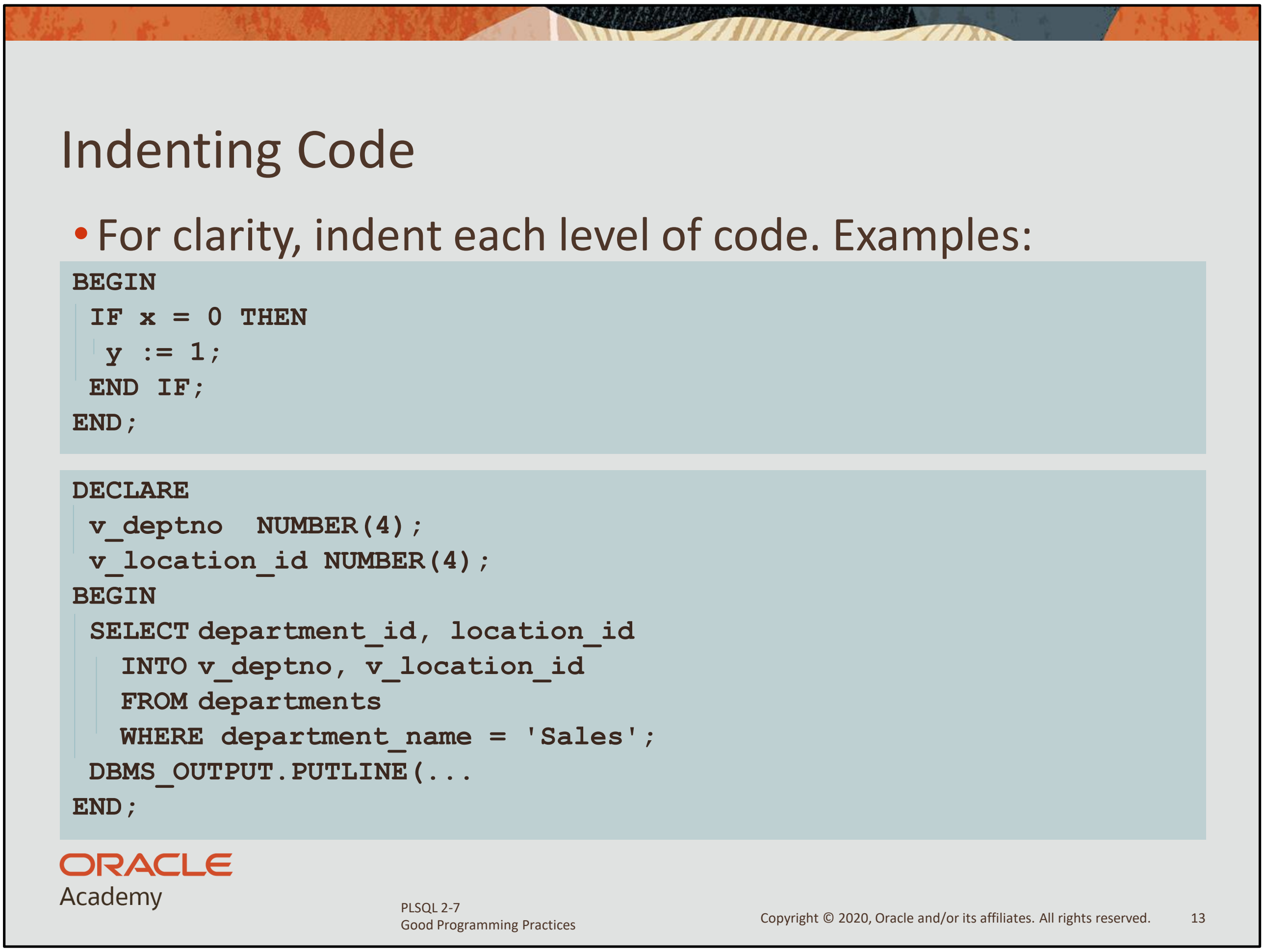
- The naming of identifiers should be clear, consistent, and unambiguous
- One commonly-used convention is to name:
  - Variables starting with v_
  - Constants starting with c_
  - Parameters starting with p_ (for passing to procedures and functions)

Naming Conventions

- Examples:
  - v_date_of_birth
  - v_last_name
  - c_tax_rate
  - c_commission_rate
  - p_employee_id
  - p_salary



ORACLE
Academy



Indenting Code

- For clarity, indent each level of code. Examples:

```
BEGIN
 IF x = 0 THEN
  y := 1;
 END IF;
END;
```

```
DECLARE
 v_deptno   NUMBER(4);
 v_location_id NUMBER(4);
BEGIN
 SELECT department_id, location_id
   INTO v_deptno, v_location_id
   FROM departments
   WHERE department_name = 'Sales';
 DBMS_OUTPUT.PUTLINE(...
END;
```

ORACLE
Academy

PLSQL 2-7
Good Programming Practices

For clarity, and to enhance readability, indent each level of code using spaces or tabs, put a space before and after operators, and begin new clauses on a new line by using carriage returns. Compare the following IF statements for readability:

OPTION #1:

```
BEGIN IF x>y THEN v_max:=x;ELSE v_max:=y;END IF;END;
```

OPTION #2:

```
BEGIN
  IF x > y THEN
    v_max := x;
   ELSE
    v_max := y;
  END IF;
END;
```

With Option #2, it is much easier to see what is happening in the code. Depending on the code editor you are using, each new level of code should be indented two spaces or one tab stop.

Summary

- In this lesson, you should have learned how to:
  - List examples of good programming practices
  - Accurately insert comments into PL/SQL code
  - Create PL/SQL code that follows formatting guidelines to produce readable code

ORACLE Academy