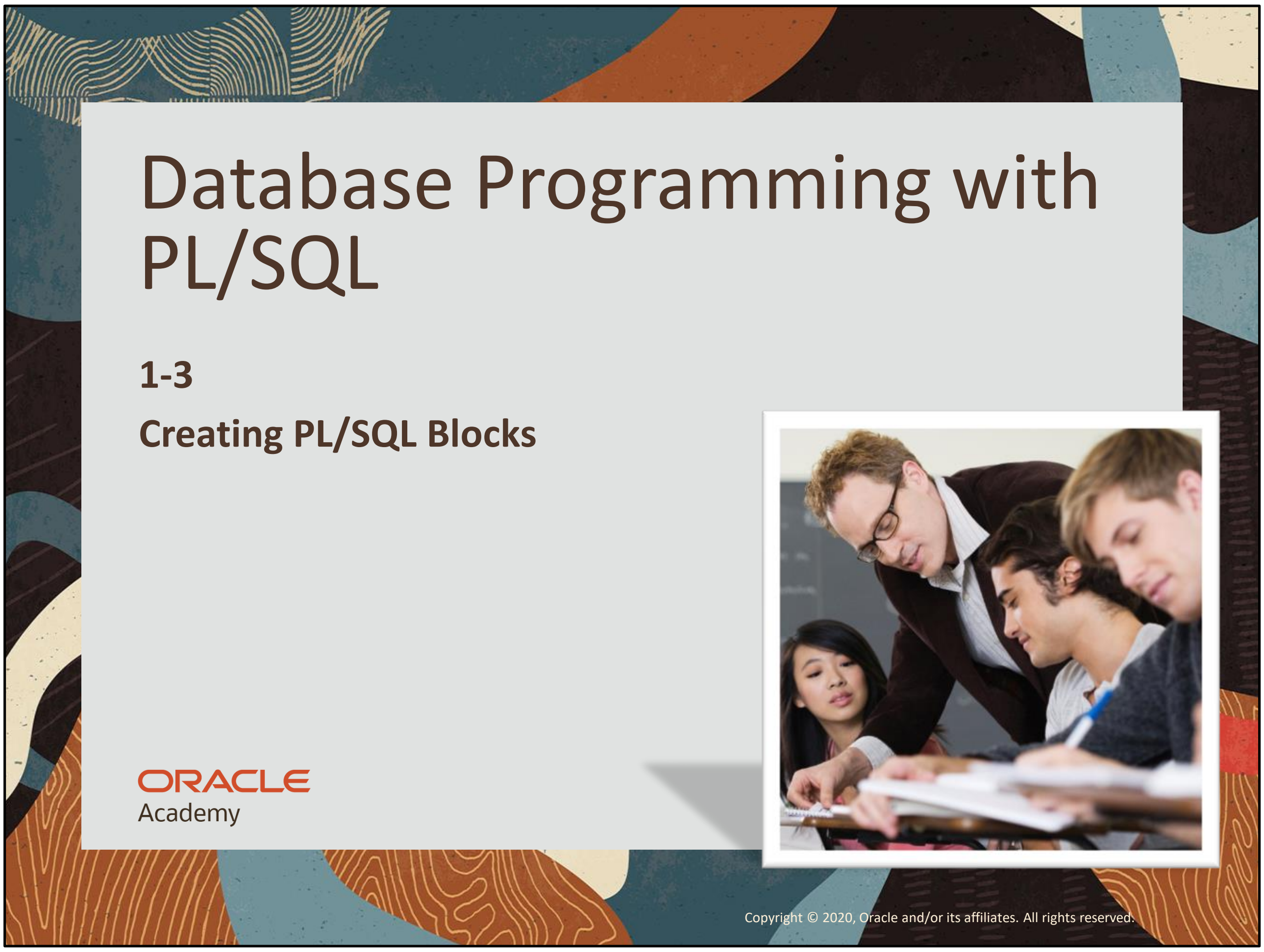
ORACLE
Academy



Database Programming with PL/SQL

1-3

Creating PL/SQL Blocks

ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Objectives

- This lesson covers the following objectives:
  - Describe the structure of a PL/SQL block
  - Identify the different types of PL/SQL blocks
  - Identify PL/SQL programming environments
  - Create and execute an anonymous PL/SQL block
  - Output messages in PL/SQL

ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Purpose

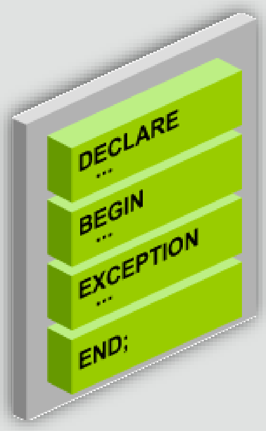
- Here you will learn the structure of a PL/SQL block and create one kind of block: an anonymous block
- Later in the course, you will learn to create Procedures, Functions, and Packages using the basic structure found in anonymous blocks
- After learning about the different environments into which you can develop your PL/SQL programs, you will also begin coding PL/SQL in the Application Express development environment

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

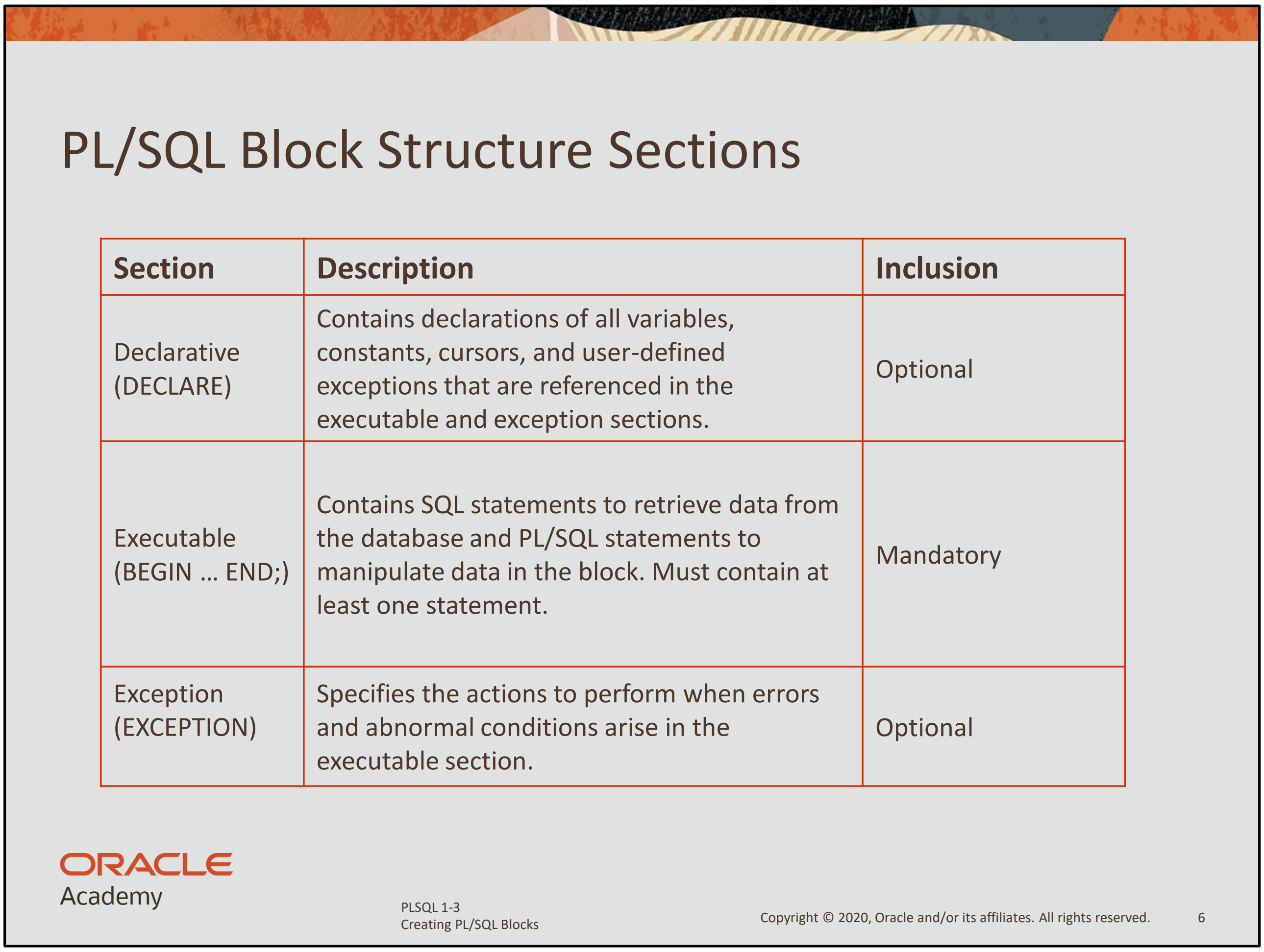
PL/SQL Block Structure

- A PL/SQL block consists of three sections

Section	Description
Declarative (optional)	The declarative section begins with the keyword DECLARE and ends when your executable section starts.
Executable (mandatory)	The executable section begins with the keyword BEGIN and ends with END. Observe that END is terminated with a semicolon. The executable section of a PL/SQL block can include any number of nested PL/SQL blocks.
Exception handling (optional)	The exception section is nested within the executable section. This section begins with the keyword EXCEPTION.



Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



PL/SQL Block Structure Sections

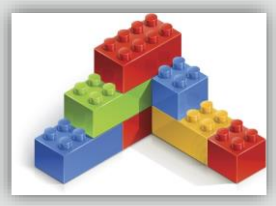
Section	Description	Inclusion
Declarative (DECLARE)	Contains declarations of all variables, constants, cursors, and user-defined exceptions that are referenced in the executable and exception sections.	Optional
Executable (BEGIN … END;)	Contains SQL statements to retrieve data from the database and PL/SQL statements to manipulate data in the block. Must contain at least one statement.	Mandatory
Exception (EXCEPTION)	Specifies the actions to perform when errors and abnormal conditions arise in the executable section.	Optional

ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

DECLARE is not needed if no variables, constants, cursors or user-defined exceptions are required. But nearly all real-life blocks will need variables and/or cursors, therefore nearly all real-life blocks will need a DECLARE section.

Anonymous Blocks



- Characteristics of anonymous blocks:
  - Unnamed block
  - Not stored in the database
  - Declared inline at the point in an application where it is executed
  - Compiled each time the application is executed
  - Passed to the PL/SQL engine for execution at run time
  - Cannot be invoked or called because it does not have a name and does not exist after it is executed

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

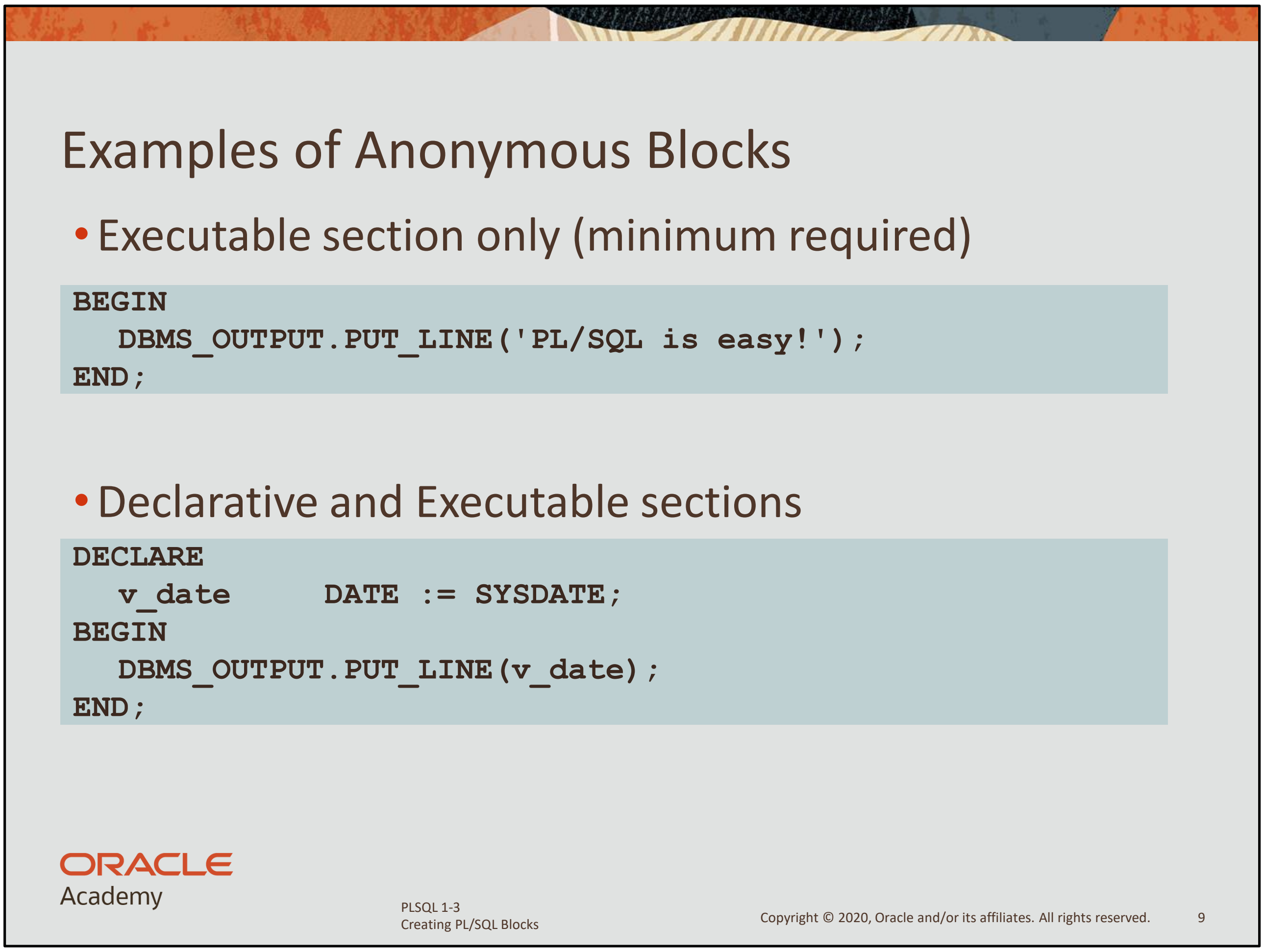
Anonymous Blocks – Basic Structure

- Basic structure of an anonymous block:

```
[DECLARE]

BEGIN
  --statements

[EXCEPTION]

END;
```

- The DECLARE and EXCEPTION keywords/sections are optional

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



Examples of Anonymous Blocks

- Executable section only (minimum required)

```
BEGIN
   DBMS_OUTPUT.PUT_LINE('PL/SQL is easy!');
END;
```

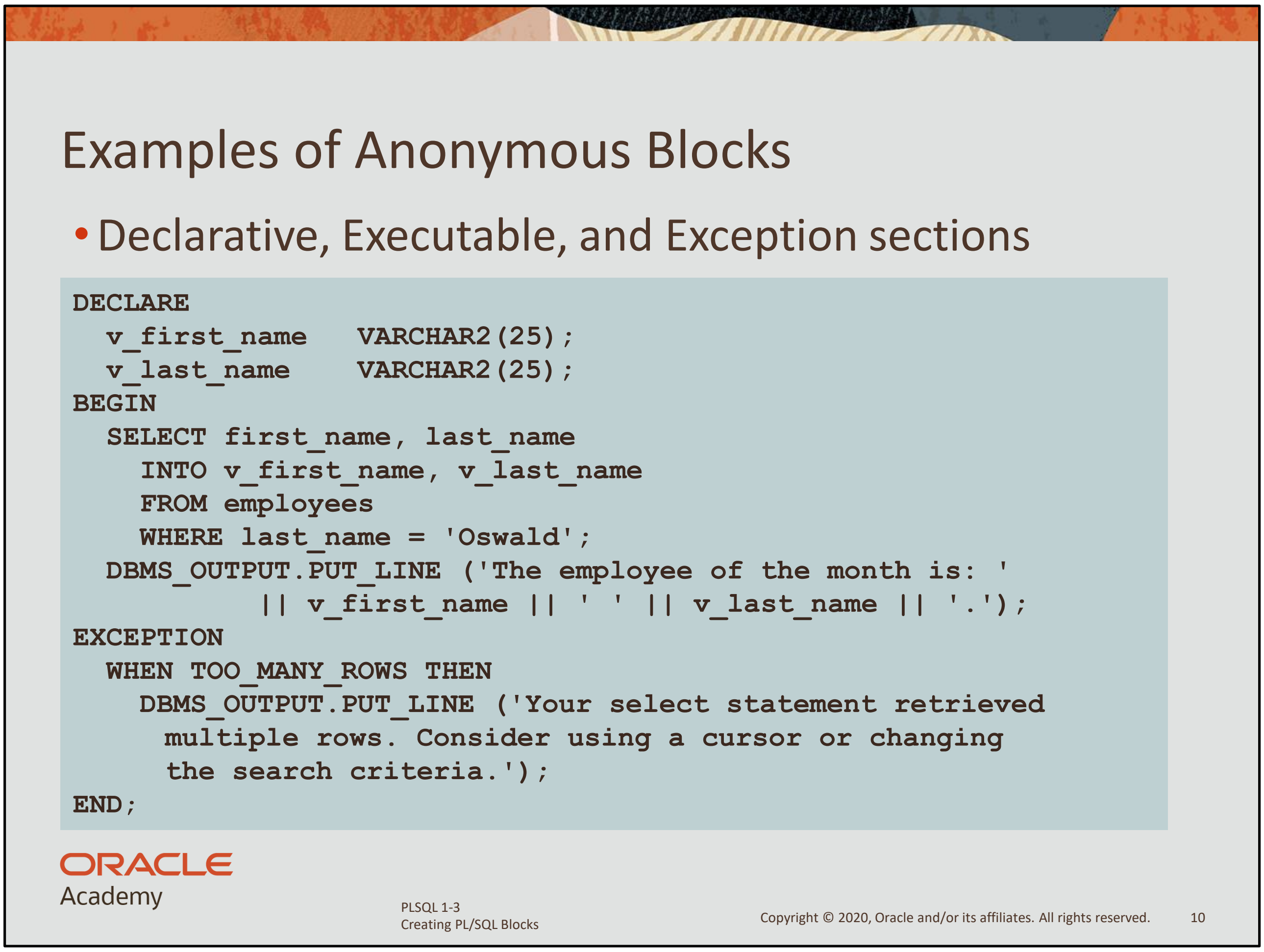
- Declarative and Executable sections

```
DECLARE
   v_date      DATE := SYSDATE;
BEGIN
   DBMS_OUTPUT.PUT_LINE(v_date);
END;
```

ORACLE Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

PUT_LINE is a function in the DBMS_OUTPUT package that displays its argument (in this slide, the value stored in v_date) on the screen for the user to see.



Examples of Anonymous Blocks

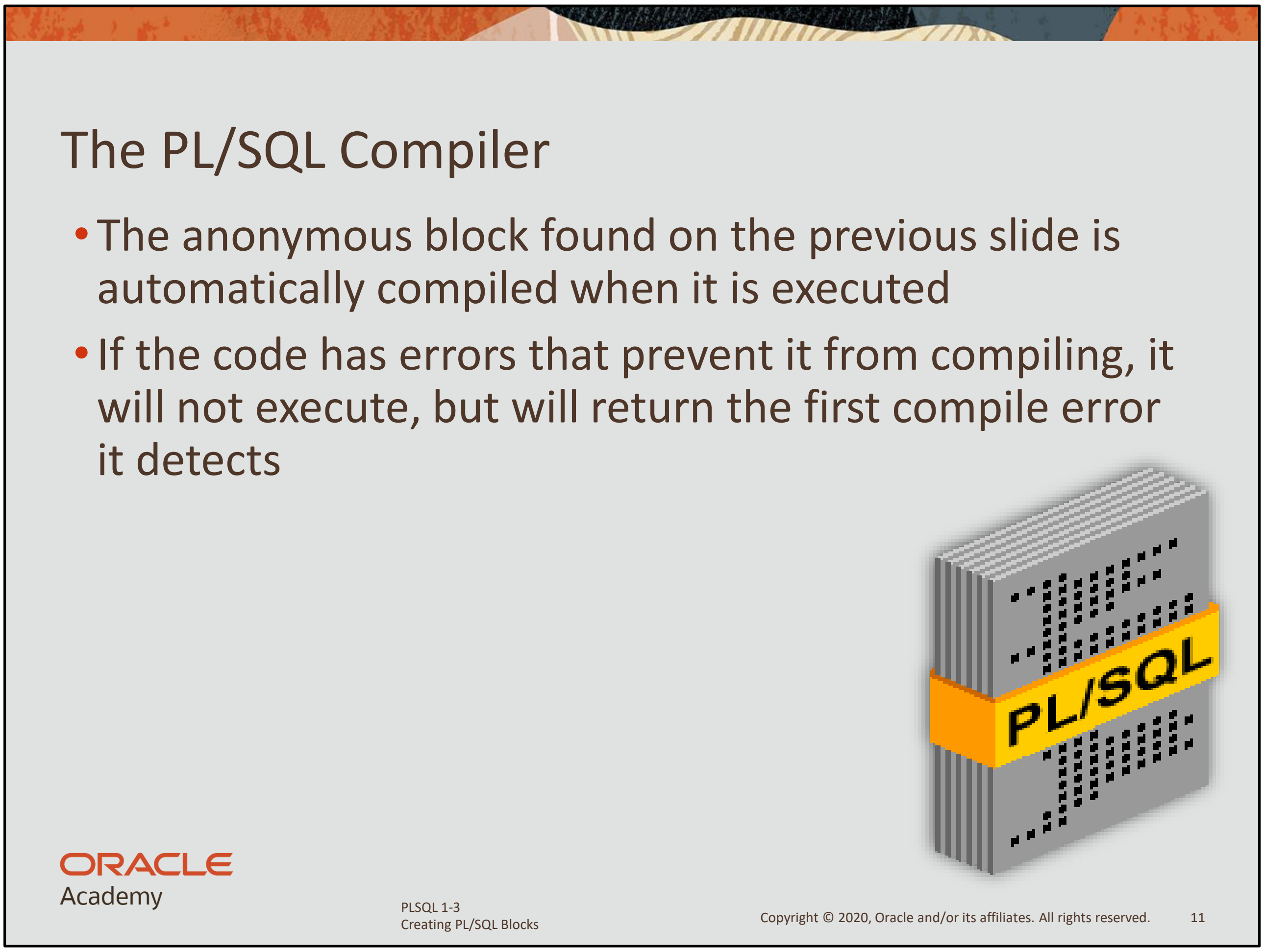
- Declarative, Executable, and Exception sections

```
DECLARE
  v_first_name   VARCHAR2(25);
  v_last_name    VARCHAR2(25);
BEGIN
  SELECT first_name, last_name
    INTO v_first_name, v_last_name
    FROM employees
    WHERE last_name = 'Oswald';
  DBMS_OUTPUT.PUT_LINE ('The employee of the month is: '
           || v_first_name || ' ' || v_last_name || '.');
EXCEPTION
  WHEN TOO_MANY_ROWS THEN
    DBMS_OUTPUT.PUT_LINE ('Your select statement retrieved
      multiple rows. Consider using a cursor or changing
      the search criteria.');
END;
```

ORACLE
Academy

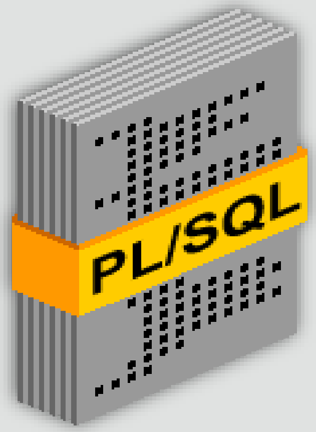
Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

What happens if the SELECT statement finds more than one employee with that last name? There is only room for one first name and one last name. Without the EXCEPTION section, the DECLARATIVE section would abort and return an Oracle error message. With the EXCEPTION section, the abort is handled by the code in the EXCEPTION section and the user (or later, the calling application) sees a friendlier error message.



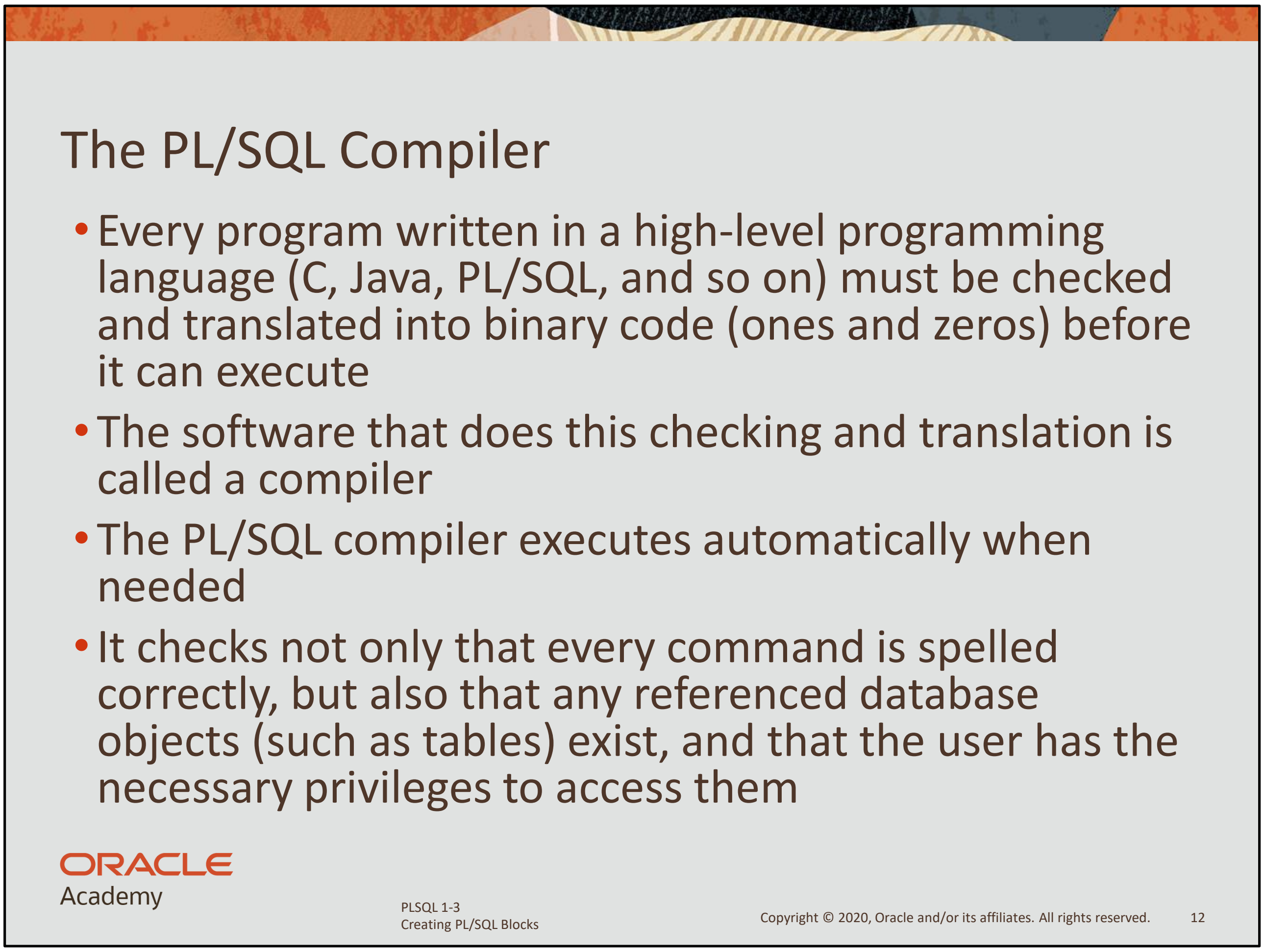
The PL/SQL Compiler

- The anonymous block found on the previous slide is automatically compiled when it is executed
- If the code has errors that prevent it from compiling, it will not execute, but will return the first compile error it detects



ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



The PL/SQL Compiler

- Every program written in a high-level programming language (C, Java, PL/SQL, and so on) must be checked and translated into binary code (ones and zeros) before it can execute
- The software that does this checking and translation is called a compiler
- The PL/SQL compiler executes automatically when needed
- It checks not only that every command is spelled correctly, but also that any referenced database objects (such as tables) exist, and that the user has the necessary privileges to access them

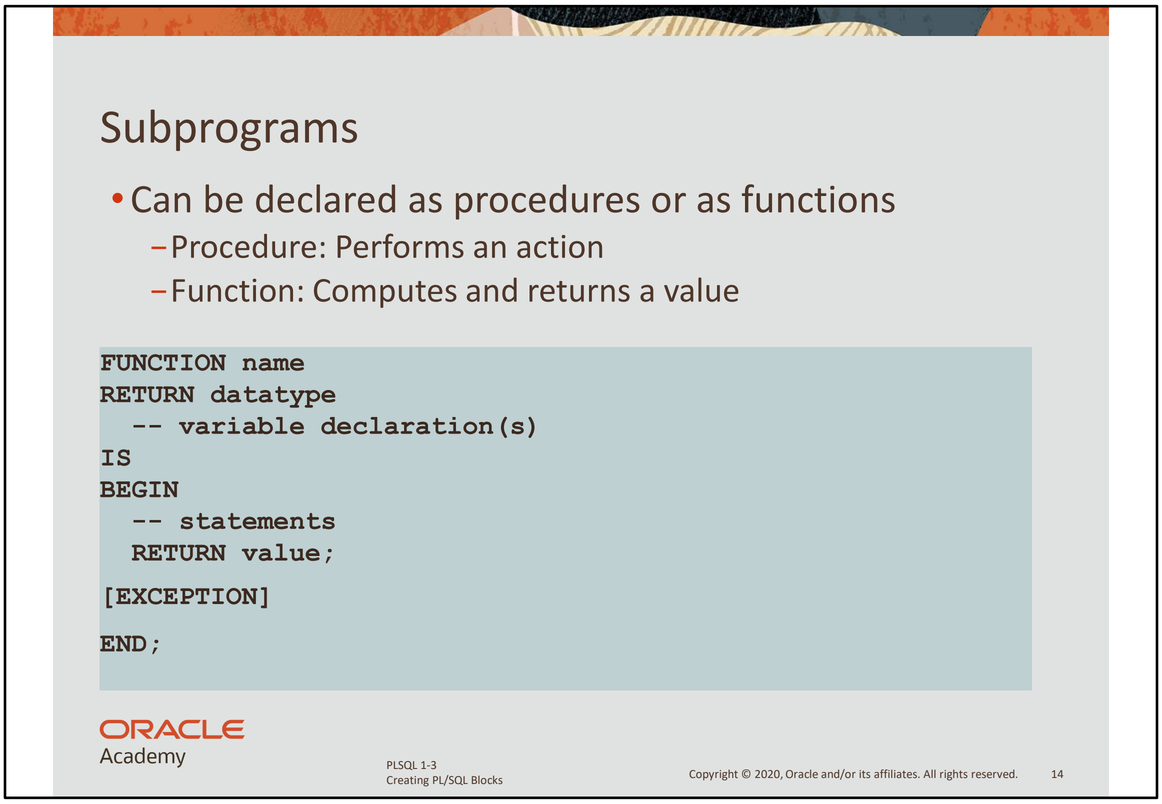
Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Subprograms

- Are named PL/SQL blocks
- Are stored in the database
- Can be invoked whenever you want depending on your application

```
PROCEDURE name
IS
  -- variable declarations
BEGIN
  -- statements
[EXCEPTION]
END;
```

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



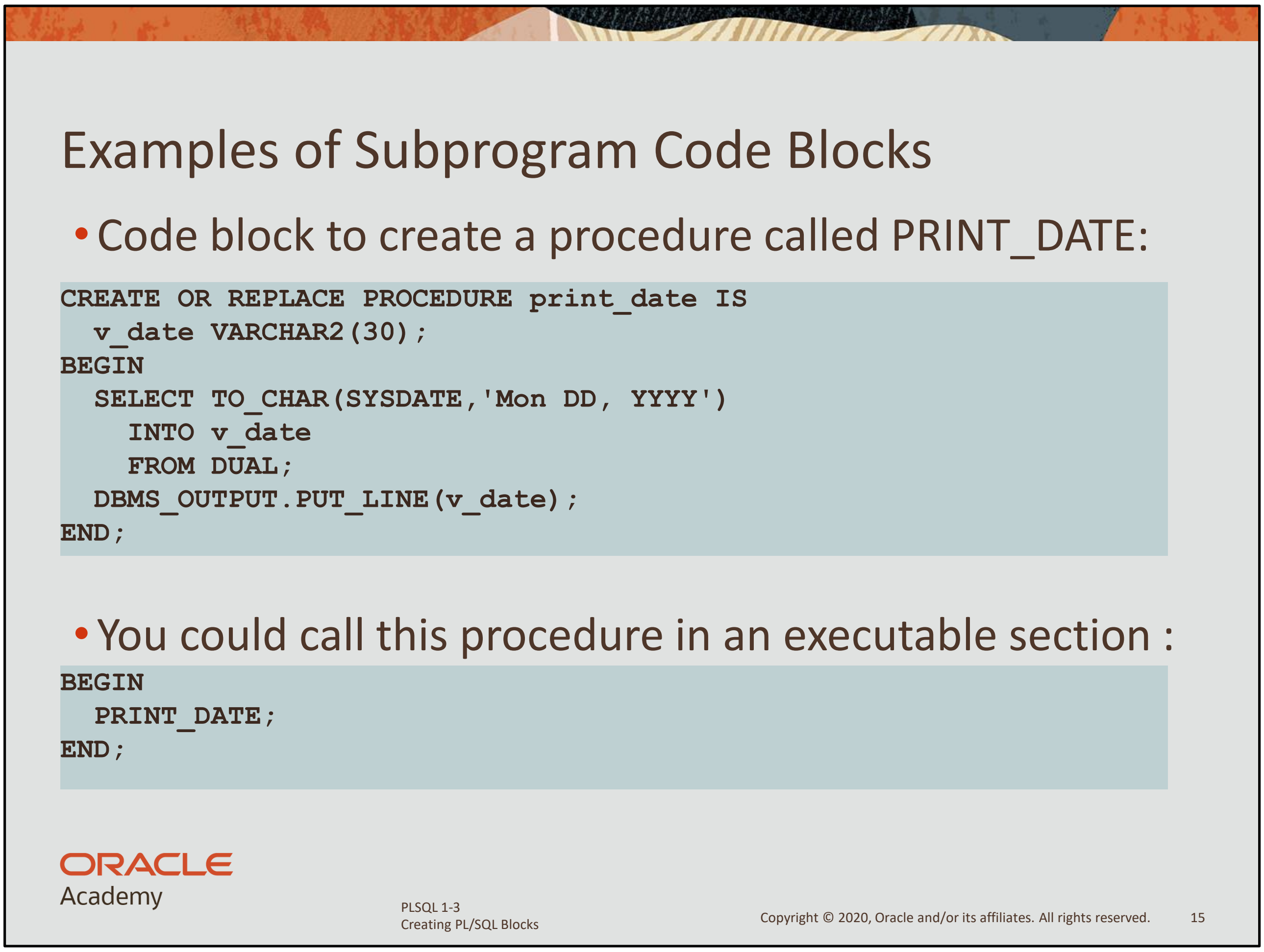
Subprograms

- Can be declared as procedures or as functions
  - Procedure: Performs an action
  - Function: Computes and returns a value

```
FUNCTION name
RETURN datatype
  -- variable declaration(s)
IS
BEGIN
  -- statements
  RETURN value;
[EXCEPTION]
END;
```

ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



Examples of Subprogram Code Blocks

- Code block to create a procedure called PRINT_DATE:

```
CREATE OR REPLACE PROCEDURE print_date IS
  v_date VARCHAR2(30);
BEGIN
  SELECT TO_CHAR(SYSDATE,'Mon DD, YYYY')
    INTO v_date
    FROM DUAL;
  DBMS_OUTPUT.PUT_LINE(v_date);
END;
```

- You could call this procedure in an executable section :

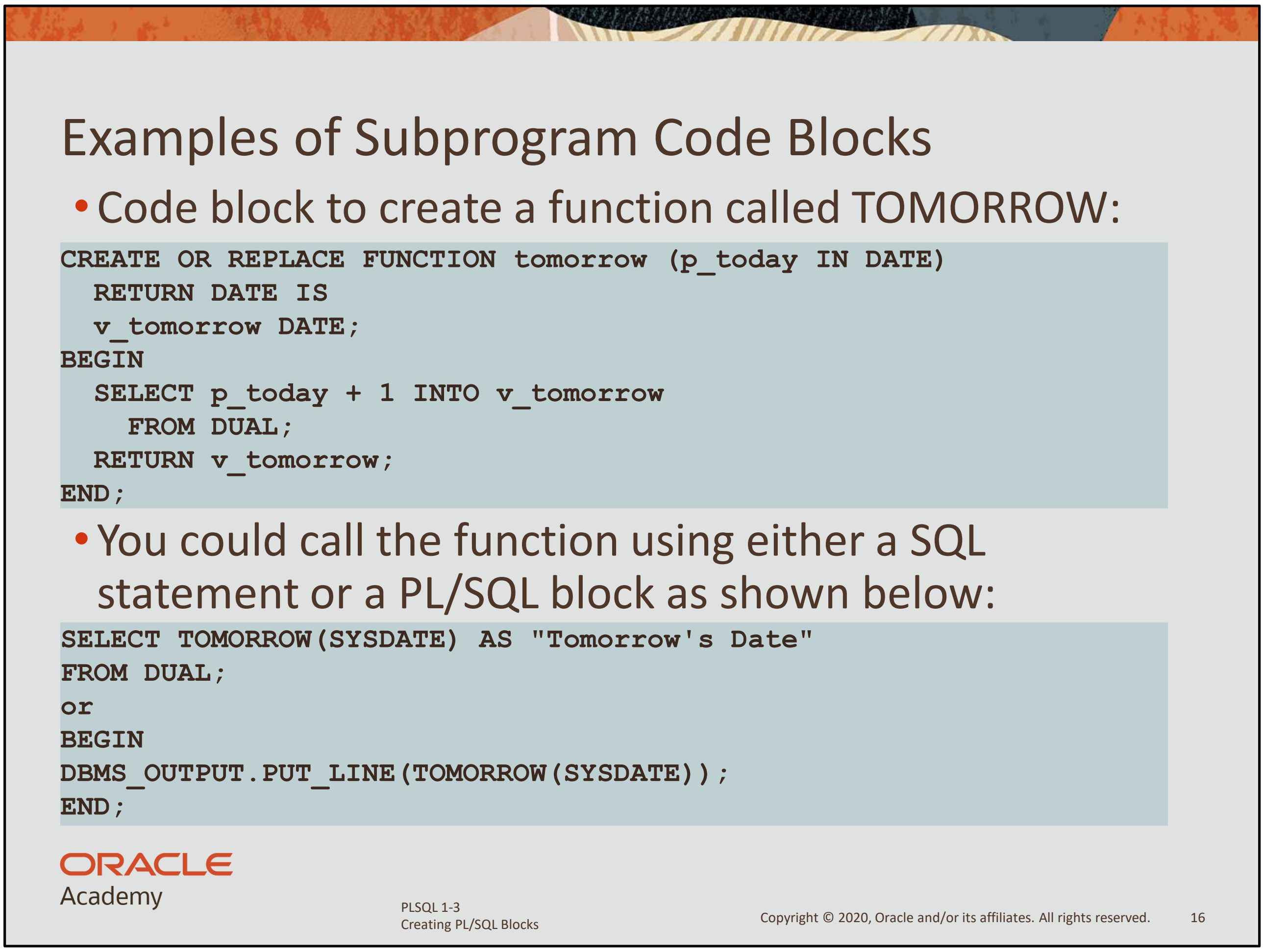
```
BEGIN
  PRINT_DATE;
END;
```

ORACLE Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

This is an example of a PL/SQL code block that is NOT an anonymous block.

This block creates a PROCEDURE that when called will display today's date.



Examples of Subprogram Code Blocks

- Code block to create a function called TOMORROW:

```
CREATE OR REPLACE FUNCTION tomorrow (p_today IN DATE)
  RETURN DATE IS
  v_tomorrow DATE;
BEGIN
  SELECT p_today + 1 INTO v_tomorrow
    FROM DUAL;
  RETURN v_tomorrow;
END;
```

- You could call the function using either a SQL statement or a PL/SQL block as shown below:

```
SELECT TOMORROW(SYSDATE) AS "Tomorrow's Date"
FROM DUAL;
or
BEGIN
DBMS_OUTPUT.PUT_LINE(TOMORROW(SYSDATE));
END;
```

ORACLE Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

This is an example of a PL/SQL code block that is NOT an anonymous block.

This block creates a FUNCTION that will return tomorrow's date (much like SYSDATE is a function that returns today's date).

The FUNCTION could also be used to initialize a variable:

```
DECLARE    v_tomorrow    DATE := TOMORROW(SYSDATE);
BEGIN
DBMS_OUTPUT.PUT_LINE('Tomorrow is ' || v_tomorrow);
END;
```

PL/SQL Programming Environments

- There are many tools available from Oracle that provide an environment for developing database-driven applications using PL/SQL

ORACLE

Application Express	Browser-based, database-driven, application development environment.
SQL Workshop	A component of Application Express.
Application Builder	A component of Application Express.
SQL Developer	An IDE for database development and management.
JDeveloper	An IDE for Java-based development.
NetBeans	An IDE for Java, HTML5, PHP, and C++.

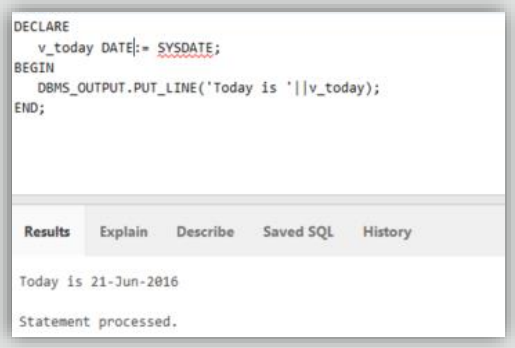
Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

This course will focus on Application Express and its SQL Workshop. You also will use Application Builder in a project. The other tools are all free downloads with excellent documentation available from www.oracle.com. All of these tools are widely-used in the business world.

SQL Commands

- As you did in the SQL course, you can use SQL Commands to enter and run a SINGLE SQL statement

- You also use SQL Commands to enter and run a SINGLE PL/SQL block

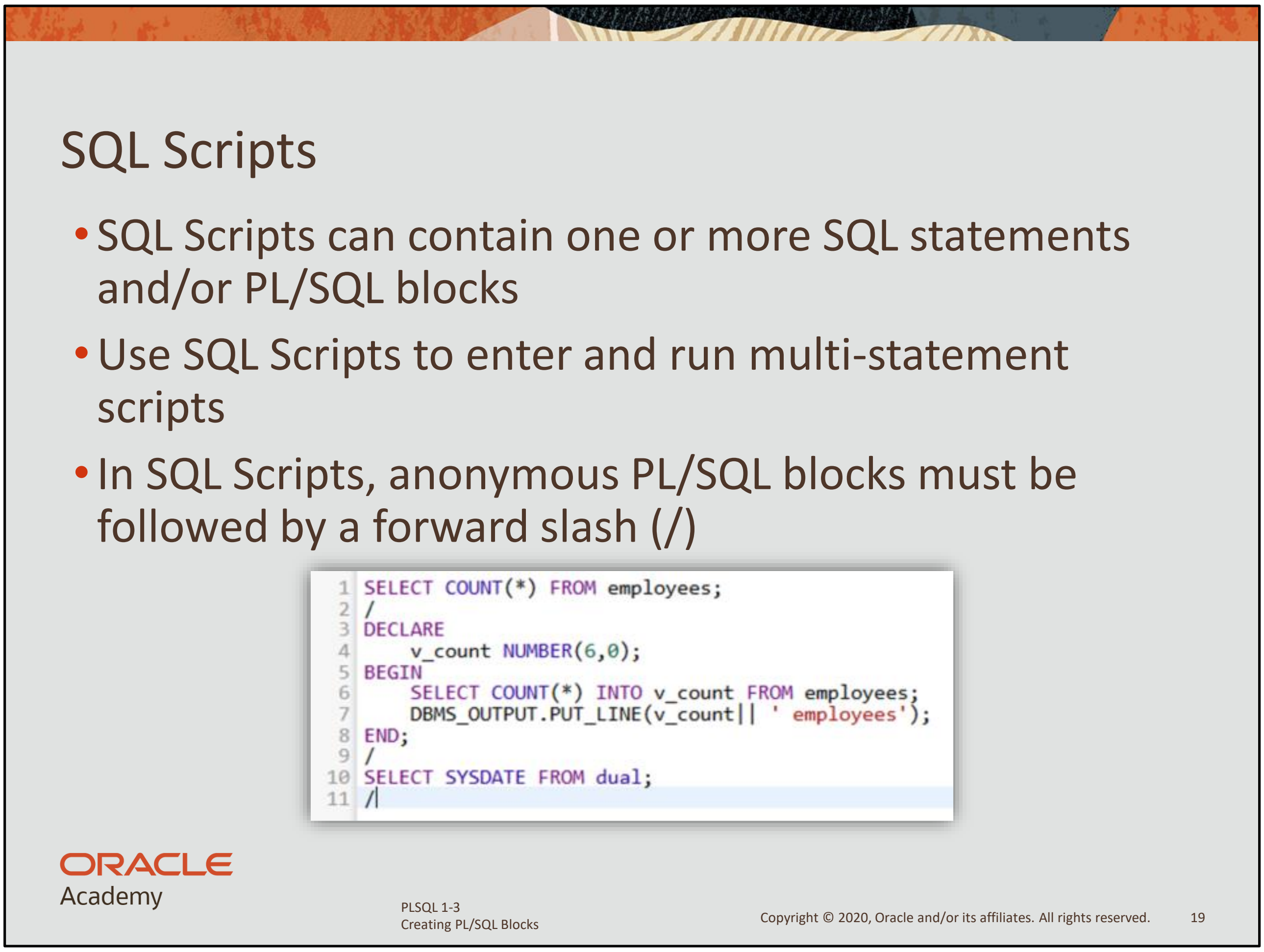


```
DECLARE
   v_today DATE:= SYSDATE;
BEGIN
   DBMS_OUTPUT.PUT_LINE('Today is '||v_today);
END;
```

```
Today is 21-Jun-2016

Statement processed.
```

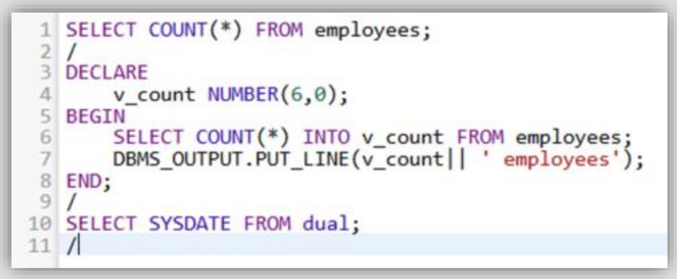
ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



SQL Scripts

- SQL Scripts can contain one or more SQL statements and/or PL/SQL blocks
- Use SQL Scripts to enter and run multi-statement scripts
- In SQL Scripts, anonymous PL/SQL blocks must be followed by a forward slash (/)



```
SELECT COUNT(*) FROM employees;
/
DECLARE
     v_count NUMBER(6,0);
BEGIN
     SELECT COUNT(*) INTO v_count FROM employees;
     DBMS_OUTPUT.PUT_LINE(v_count|| ' employees');
END;
/
SELECT SYSDATE FROM dual;
/
```

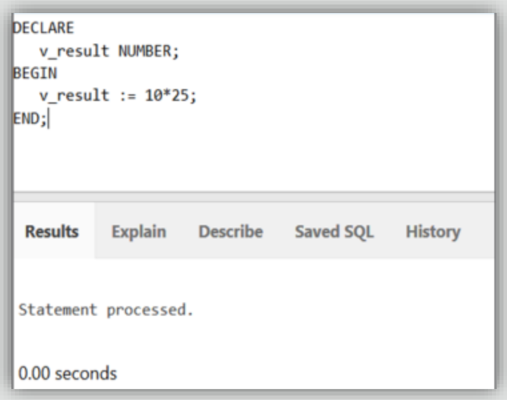
ORACLE Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Although not required, the convention is to follow all SQL statements with a forward slash (/), as well as the anonymous blocks, in SQL script files.

Using DBMS_OUTPUT.PUT_LINE Example

- Look at this simple PL/SQL block and its output
- How can you display the result?



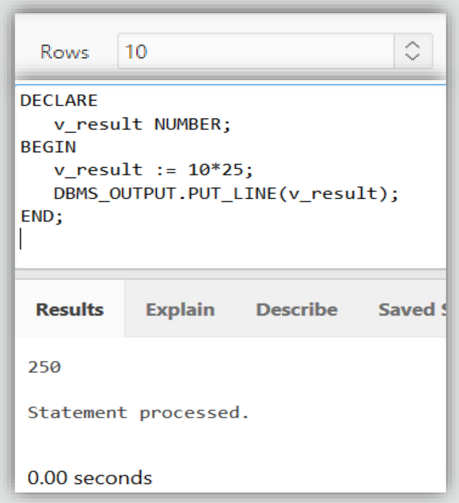
```
DECLARE
   v_result NUMBER;
BEGIN
   v_result := 10*25;
END;
```

```
Statement processed.

0.00 seconds
```

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Using DBMS_OUTPUT.PUT_LINE Example

- Let's add a call to the PUT_LINE function in the DBMS_OUTPUT package.
- Now you can see the result!

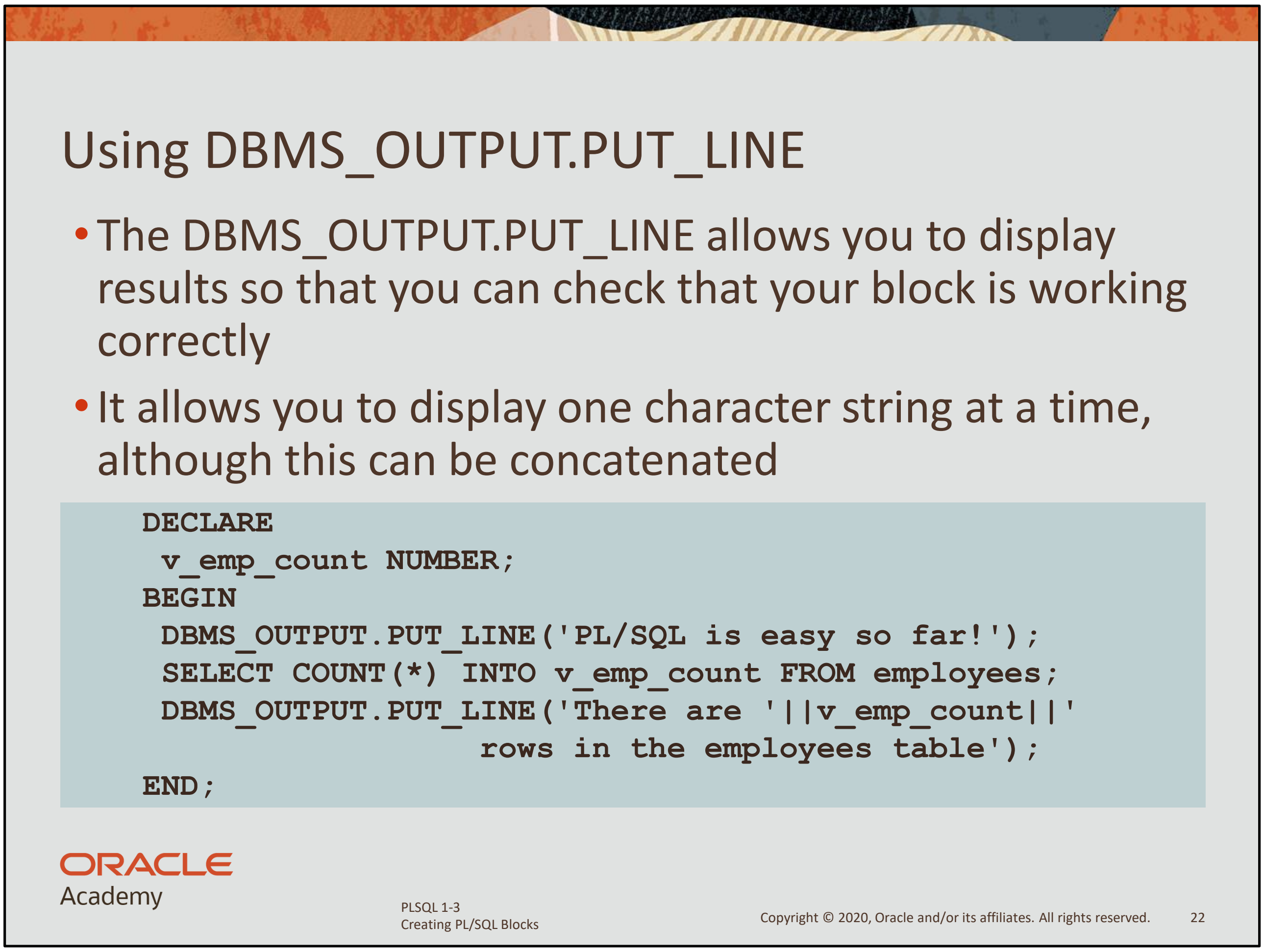


```
DECLARE
   v_result NUMBER;
BEGIN
   v_result := 10*25;
   DBMS_OUTPUT.PUT_LINE(v_result);
END;
```

```
250

Statement processed.

0.00 seconds
```

ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



Using DBMS_OUTPUT.PUT_LINE

- The DBMS_OUTPUT.PUT_LINE allows you to display results so that you can check that your block is working correctly
- It allows you to display one character string at a time, although this can be concatenated

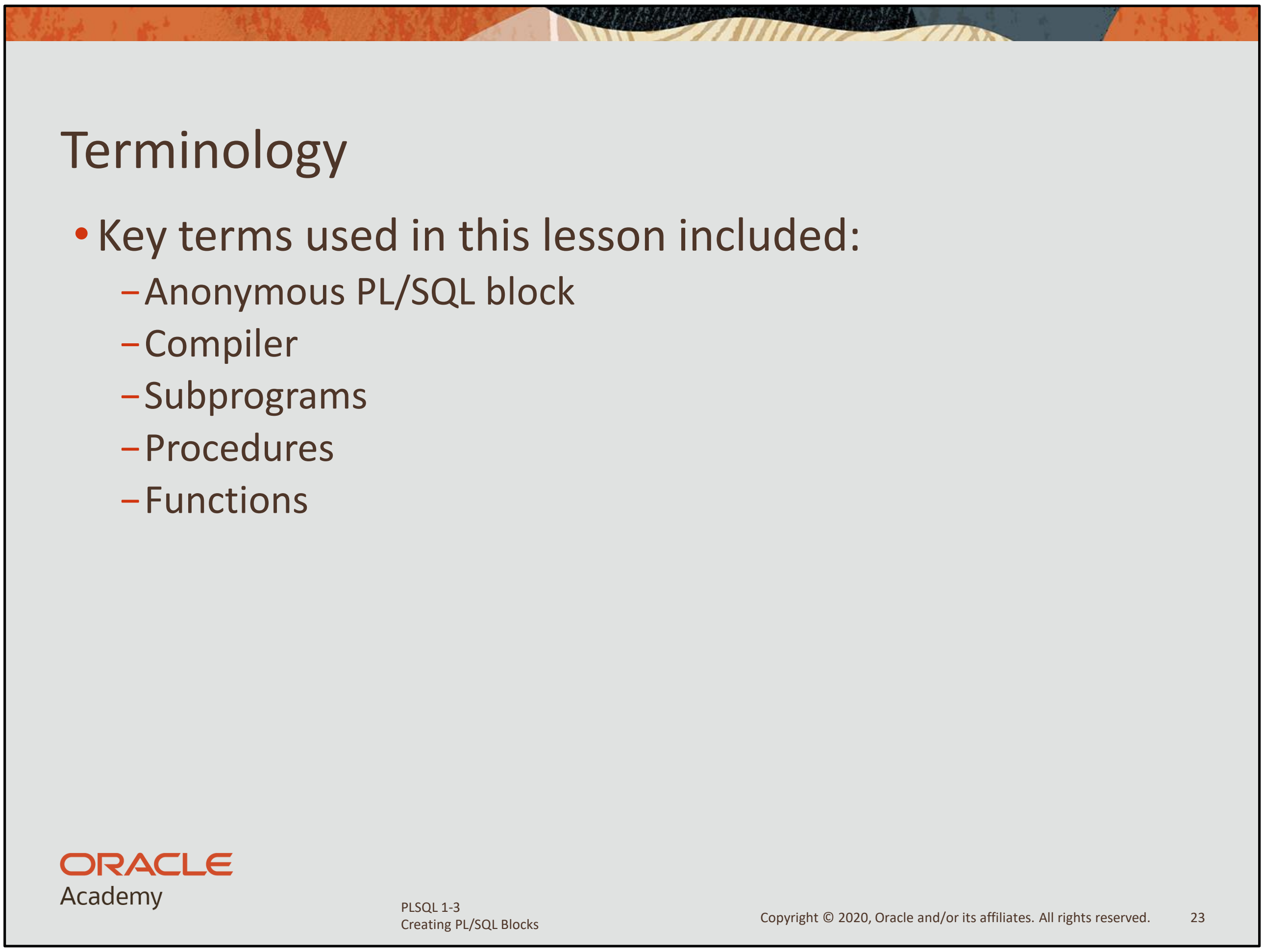
```
DECLARE
 v_emp_count NUMBER;
BEGIN
 DBMS_OUTPUT.PUT_LINE('PL/SQL is easy so far!');
 SELECT COUNT(*) INTO v_emp_count FROM employees;
 DBMS_OUTPUT.PUT_LINE('There are '||v_emp_count||'
                  rows in the employees table');
END;
```

ORACLE Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

The second DBMS_OUTPUT.PUT_LINE call in the slide shows that number values (v_emp_count) can be displayed by PUT_LINE in combination with string values.

In this example, the Oracle server has performed an implicit datatype conversion (TO_CHAR(v_emp_count)) to convert the number to a character string for concatenation.



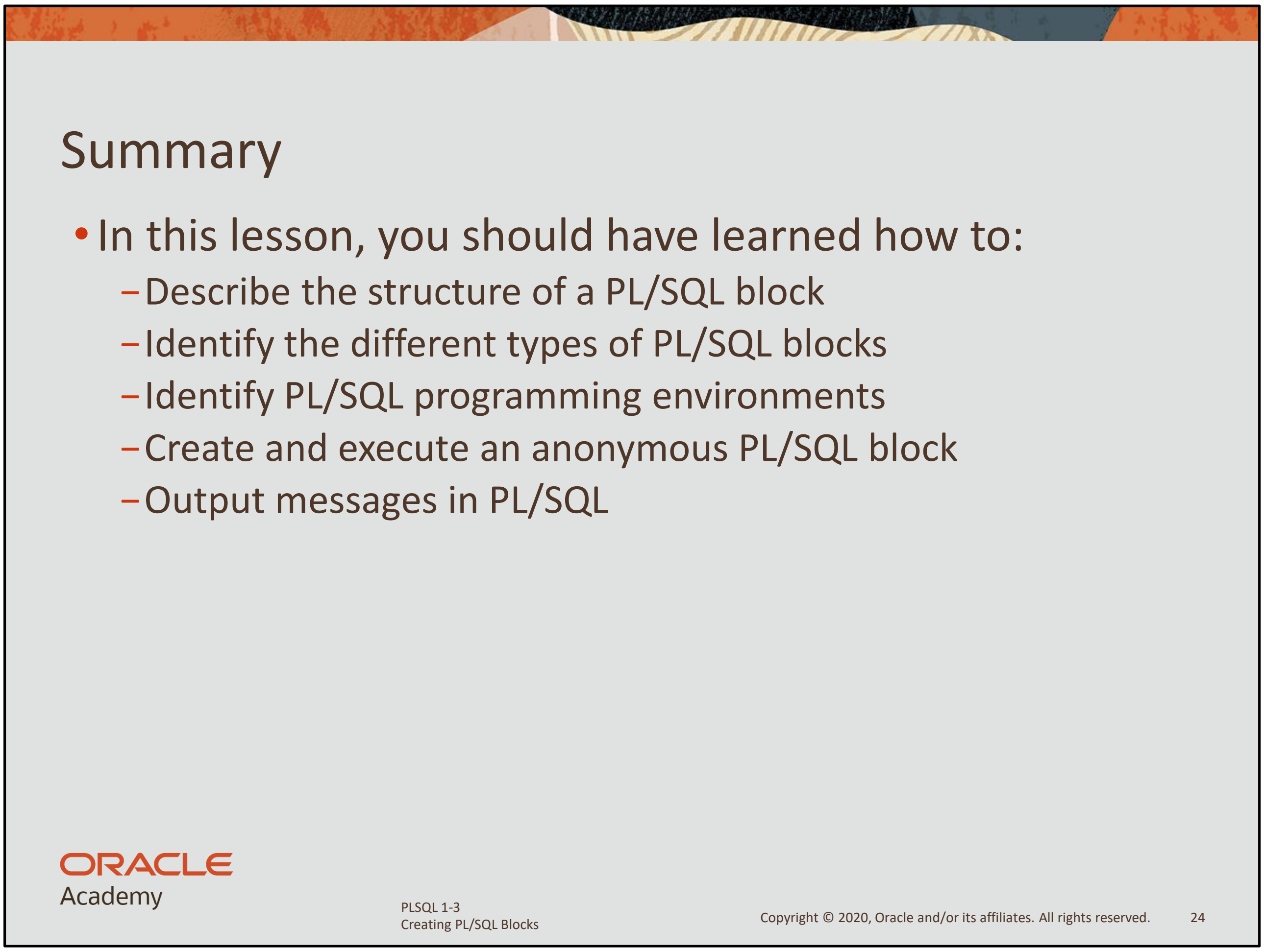
Terminology

- Key terms used in this lesson included:
  - Anonymous PL/SQL block
  - Compiler
  - Subprograms
  - Procedures
  - Functions

ORACLE
Academy

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

- Anonymous PL/SQL block – unnamed blocks of code not stored in the database and do not exist after they are executed
- Compiler – software that checks and translates programs written in high-level programming languages into binary code to execute
- Subprograms – named PL/SQL blocks that are stored in the database and can be declared as PROCEDURES or FUNCTIONS
- Procedures – subprograms that perform an action and may return one or more values
- Functions – subprograms that return a single value



Summary

- In this lesson, you should have learned how to:
  - Describe the structure of a PL/SQL block
  - Identify the different types of PL/SQL blocks
  - Identify PL/SQL programming environments
  - Create and execute an anonymous PL/SQL block
  - Output messages in PL/SQL

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

ORACLE

Academy