ORACLE
Academy

Database Programming with PL/SQL

6-2

Indexing Tables of Records

Academy

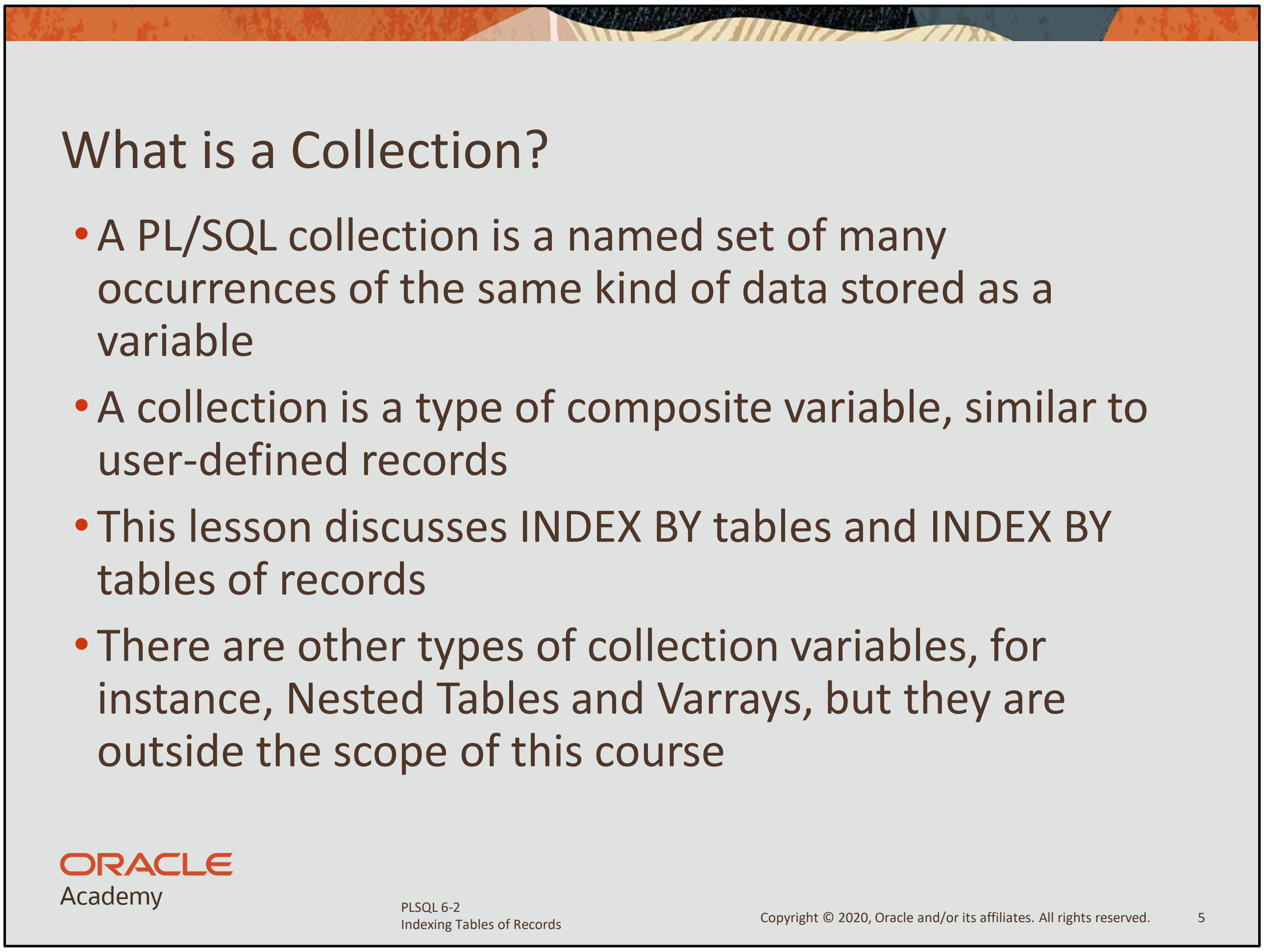
Objectives

- This lesson covers the following objectives:
  - Create an INDEX BY table
  - Create an INDEX BY table of records
  - Describe the difference between records, tables, and tables of records

Purpose

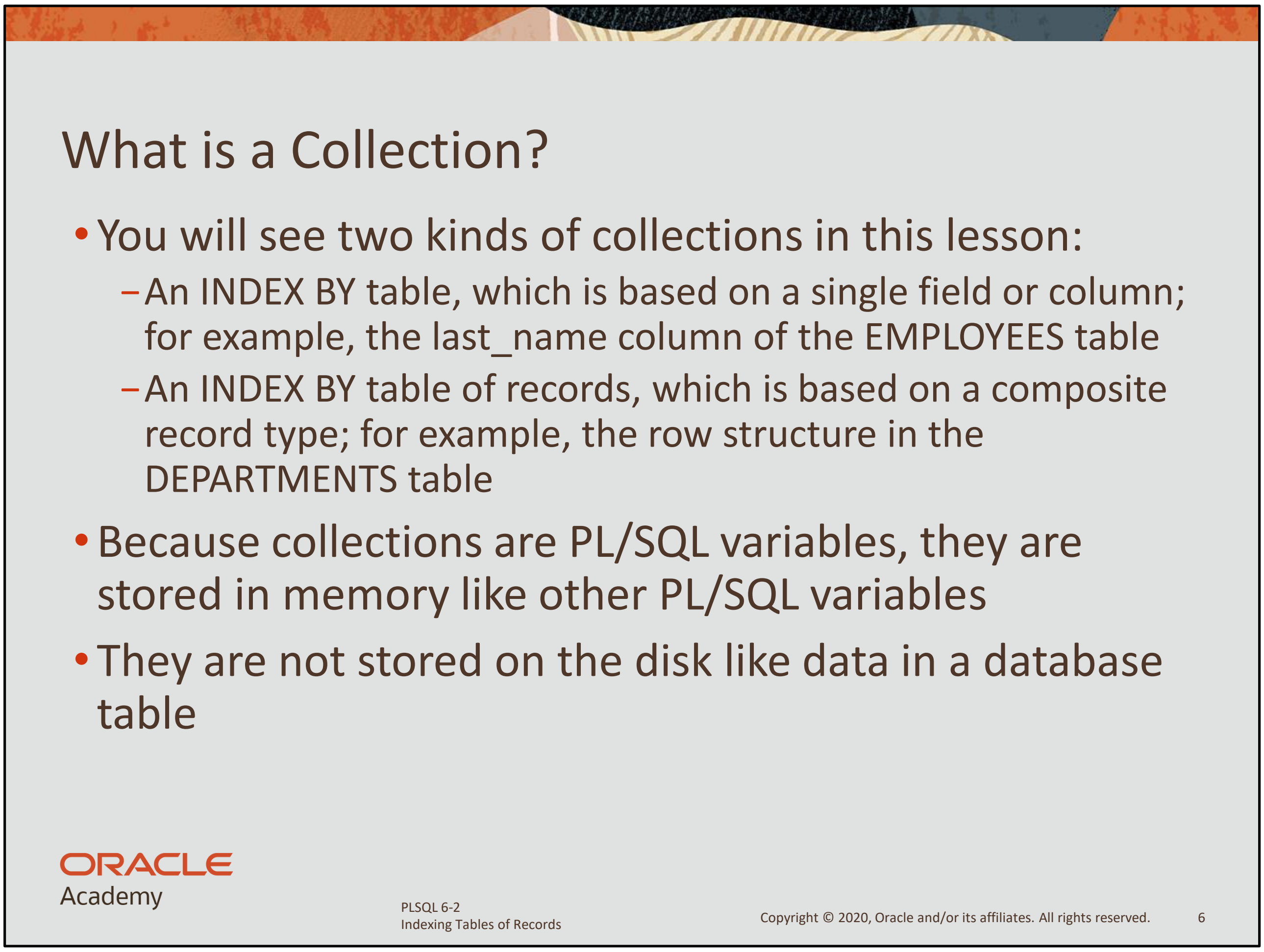
- You have learned that you can temporarily store one record in a single variable, either by using %ROWTYPE or a user-defined record
- However, there are times when you need to temporarily store multiple rows of data
- You might do this when calculating averages or accumulating items in an online shopping cart prior to checking out
- These are called collections

ORACLE
Academy



What is a Collection?

- A PL/SQL collection is a named set of many occurrences of the same kind of data stored as a variable
- A collection is a type of composite variable, similar to user-defined records
- This lesson discusses INDEX BY tables and INDEX BY tables of records
- There are other types of collection variables, for instance, Nested Tables and Varrays, but they are outside the scope of this course



What is a Collection?

- You will see two kinds of collections in this lesson:
  - An INDEX BY table, which is based on a single field or column; for example, the last_name column of the EMPLOYEES table
  - An INDEX BY table of records, which is based on a composite record type; for example, the row structure in the DEPARTMENTS table
- Because collections are PL/SQL variables, they are stored in memory like other PL/SQL variables
- They are not stored on the disk like data in a database table

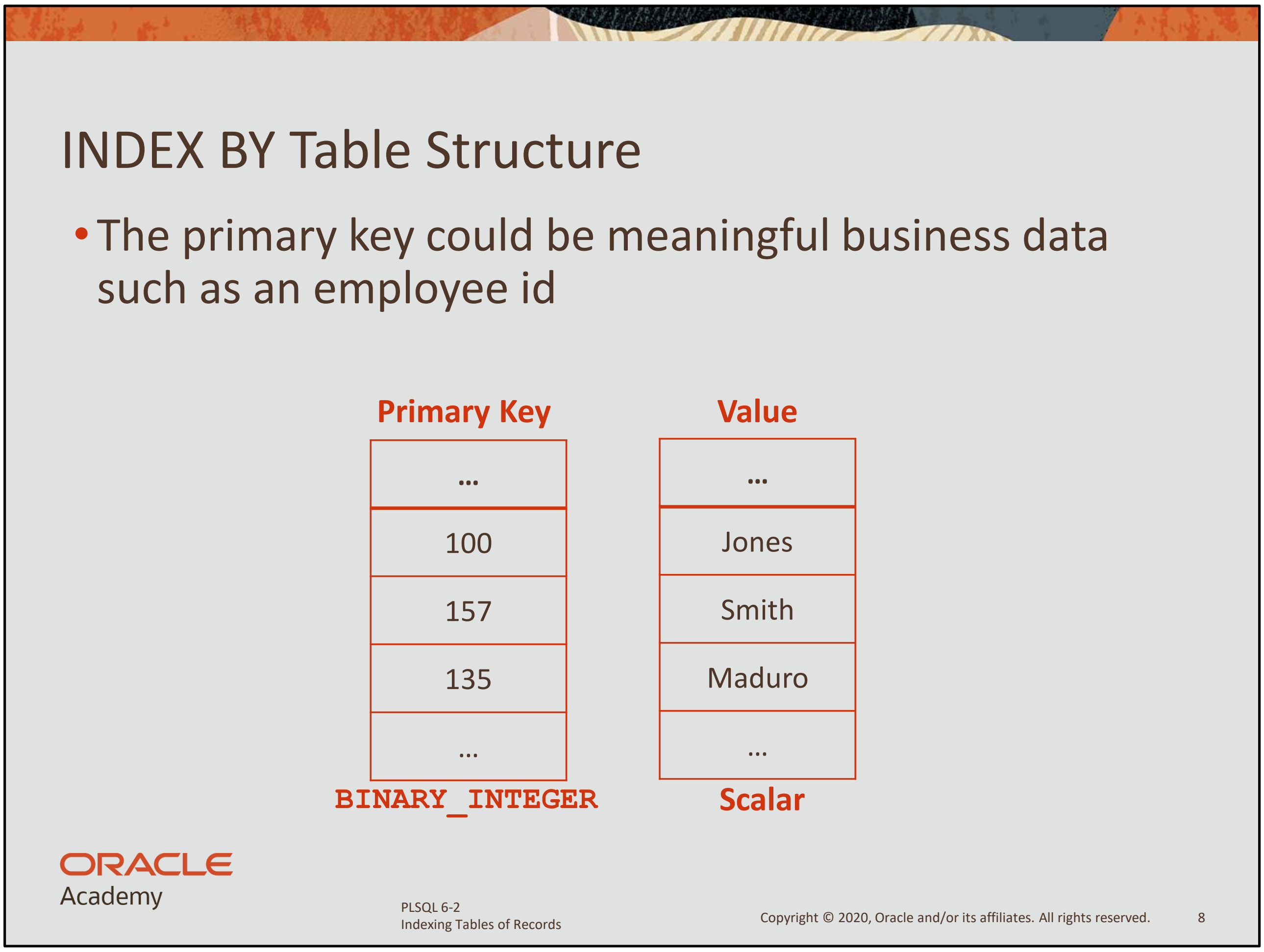
ORACLE Academy

INDEX BY tables are sometimes referred to as "Associative Arrays."

An INDEX BY Table Has a Primary Key

- We need to be able to reference each row in an INDEX BY table
- Therefore, every INDEX BY table must have a primary key which serves as an index to the data
- The primary key is typically a BINARY_INTEGER, but it may be a VARCHAR2

ORACLE
Academy



INDEX BY Table Structure

- The primary key could be meaningful business data such as an employee id

Primary Key	Value
…	…
100	Jones
157	Smith
135	Maduro
…	…
BINARY_INTEGER	Scalar

ORACLE Academy

The primary key is not like a sequence. Values are not generated automatically, but must be specifically inserted. Therefore some values can be missing, as the slide shows.

Although an INDEX BY table can have a VARCHAR2 primary key, a BINARY_INTERGER is frequently used to its faster speed and more efficient storage.

Declaring an INDEX BY Table

- Like user-defined records, you must first declare a type and then declare a variable of that type
- The syntax is:

```
TYPE type_name IS TABLE OF DATA_TYPE
  INDEX BY PRIMARY_KEY_DATA_TYPE;
identifier    type_name;
```

- The following example sets up an INDEX BY table to hold all of the hire_dates from the EMPLOYEES table

```
TYPE t_hire_date IS TABLE OF DATE
  INDEX BY BINARY_INTEGER;
v_hire_date_tab    t_hire_date;
```

ORACLE
Academy

Populating an INDEX BY Table

- The syntax to populate the INDEX BY table is:

```
DECLARE
  TYPE type_name IS TABLE OF DATA_TYPE
  INDEX BY PRIMARY_KEY_DATA_TYPE;
  identifier   type_name;
BEGIN
  FOR record IN (SELECT column FROM table)
 LOOP
   identifier(primary_key) := record.column;
  END LOOP;
END;
```

- The primary key can be initialized using a unique column from the selected table or an incrementing integer

ORACLE
Academy

Populating an INDEX BY Table

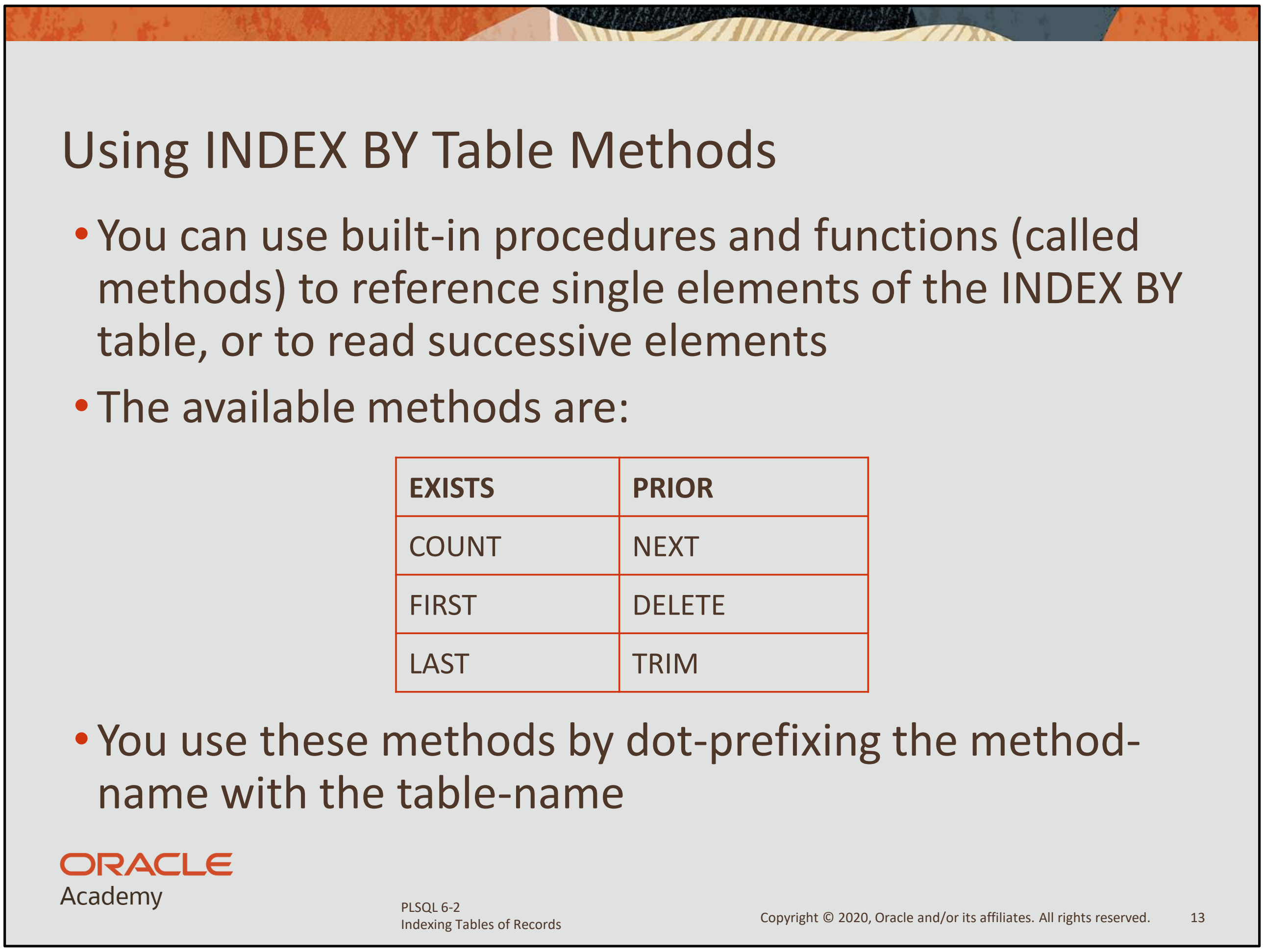
- This example populates an INDEX BY table with the hire date of employees using employee_id as the primary key

```
DECLARE
  TYPE t_hire_date IS TABLE OF employees.hire_date%TYPE
 INDEX BY BINARY_INTEGER;
  v_hire_date_tab   t_hire_date;
BEGIN
  FOR emp_rec IN
 (SELECT employee_id, hire_date FROM employees)
 LOOP
   v_hire_date_tab(emp_rec.employee_id)
  := emp_rec.hire_date;
  END LOOP;
END;
```

Populating an INDEX BY Table

- This example populates an INDEX BY table with employees' date of hire and sets the primary key using a sequence derived from incrementing v_count

```
DECLARE
  TYPE t_hire_date IS TABLE OF employees.hire_date%TYPE
 INDEX BY BINARY_INTEGER;
  v_hire_date_tab   t_hire_date;
  v_count BINARY_INTEGER := 0;
BEGIN
  FOR emp_rec IN
 (SELECT hire_date FROM employees)
 LOOP
   v_count := v_count + 1;
   v_hire_date_tab(v_count) := emp_rec.hire_date;
  END LOOP;
END;
```



Using INDEX BY Table Methods

- You can use built-in procedures and functions (called methods) to reference single elements of the INDEX BY table, or to read successive elements
- The available methods are:

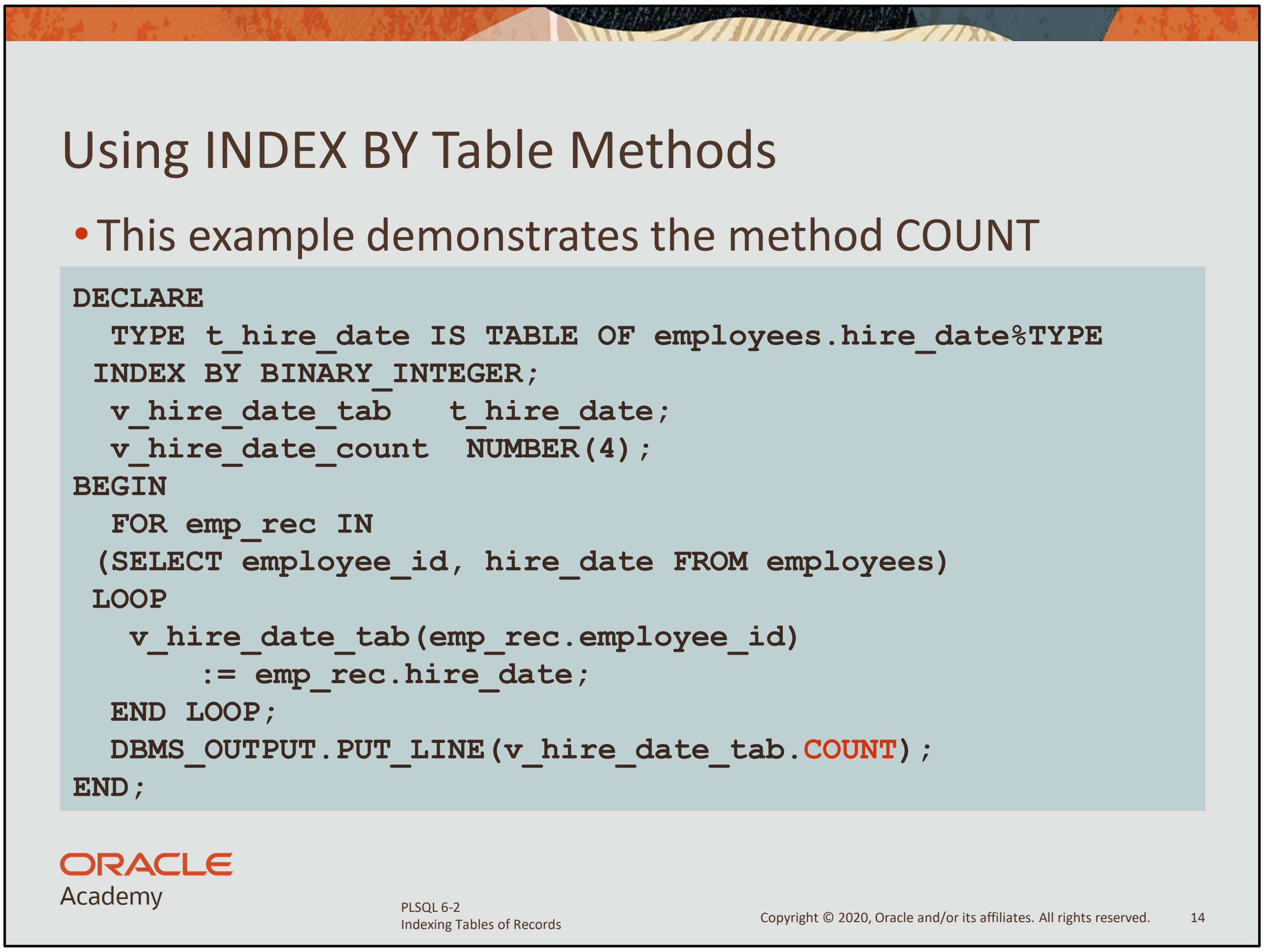
EXISTS	**PRIOR**
COUNT	NEXT
FIRST	DELETE
LAST	TRIM

- You use these methods by dot-prefixing the method-name with the table-name

ORACLE Academy

EXISTS(n), PRIOR(n), and NEXT(n) take an index number (n) corresponding to a row in the INDEX BY table as a parameter.

TRIM and DELETE also may take parameters.



Using INDEX BY Table Methods

- This example demonstrates the method COUNT

```
DECLARE
  TYPE t_hire_date IS TABLE OF employees.hire_date%TYPE
 INDEX BY BINARY_INTEGER;
  v_hire_date_tab    t_hire_date;
  v_hire_date_count  NUMBER(4);
BEGIN
  FOR emp_rec IN
 (SELECT employee_id, hire_date FROM employees)
 LOOP
   v_hire_date_tab(emp_rec.employee_id)
       := emp_rec.hire_date;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE(v_hire_date_tab.COUNT);
END;
```

ORACLE Academy

The COUNT method returns the number of elements in the INDEX BY table.

INDEX BY Table of Records

- Even though an INDEX BY table can have only one data field, that field can be a composite data type such as a RECORD
- This is an INDEX BY table of records
- The record can be %ROWTYPE or a user-defined record
- This example declares an INDEX BY table to store complete rows from the EMPLOYEES table:

```
DECLARE
  TYPE t_emp_rec IS TABLE OF employees%ROWTYPE
 INDEX BY BINARY_INTEGER;

 v_employees_tab   t_emprec;
```

INDEX BY Table of Records

- Individual fields within a table of records can be referenced by adding an index value in parentheses after the table of records name
- Syntax: table(index).field
- Example: v_employees_tab(index).hire_date
- The index value in the example could be an actual value (ex. 1, 5, 12, etc.) or a reference to a value (v_emp_rec_tab.LAST)

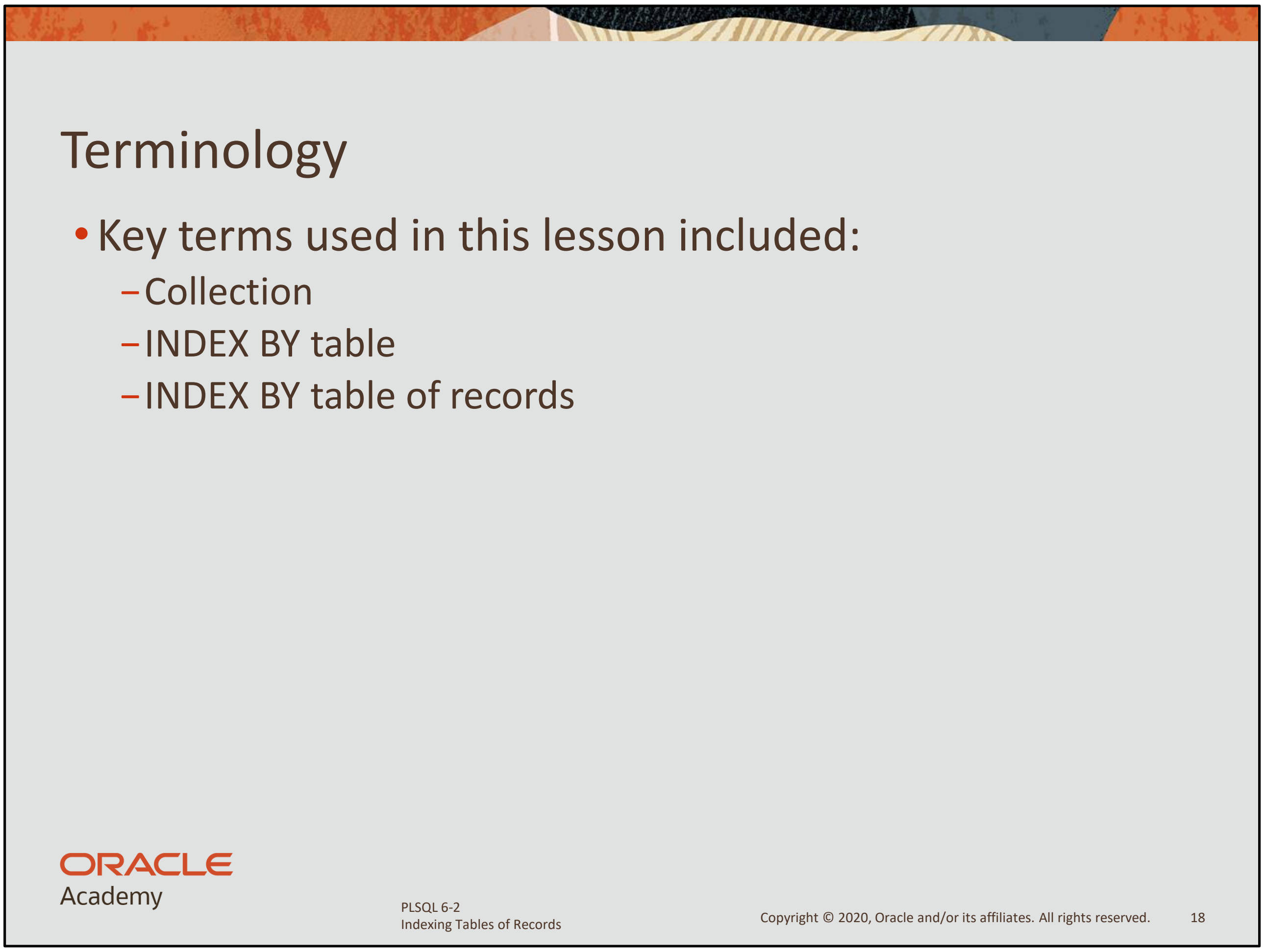
ORACLE
Academy

Using an INDEX BY Table of Records

- This example is similar to the earlier INDEX BY table example, but stores the entire EMPLOYEES row and displays the salary from each row

```
DECLARE
  TYPE t_emp_rec IS TABLE OF employees%ROWTYPE
 INDEX BY BINARY_INTEGER;
  v_emp_rec_tab   t_emp_rec;
BEGIN
  FOR emp_rec IN (SELECT * FROM employees) LOOP
 v_emp_rec_tab(emp_rec.employee_id) := emp_rec;
 DBMS_OUTPUT.PUT_LINE(
   v_emp_rec_tab(emp_rec.employee_id).salary);
  END LOOP;
END;
```

ORACLE
Academy


Terminology

- Key terms used in this lesson included:
  - Collection
  - INDEX BY table
  - INDEX BY table of records

ORACLE
Academy

- Collection – A named set of multiple occurrences of the same kind of data
- INDEX BY table – A collection which is based on a single field or column
- INDEX BY table of records – A collection which is based on a composite record type

Summary

- In this lesson, you should have learned how to:
  - Create an INDEX BY table
  - Create an INDEX BY table of records
  - Describe the difference between records, tables, and tables of records

ORACLE
Academy