Database Programming with PL/SQL

2-2

Recognizing PL/SQL Lexical Units
Objectives

This lesson covers the following objectives:

• List and define the different types of lexical units available in PL/SQL

• Describe identifiers and identify valid and invalid identifiers in PL/SQL

• Describe and identify reserved words, delimiters, literals, and comments in PL/SQL
Purpose

• A spoken language has different parts of speech.
• Each part of speech (such as an adjective, noun, and verb) is used differently and must follow rules.
• Similarly, a programming language has different parts of speech that are used differently and must follow rules.
• These parts of speech are called lexical units.
Lexical Units in a PL/SQL Block

Lexical units:
• Are the building blocks of any PL/SQL block
• Are sequences of characters including letters, digits, tabs, returns, and symbols
• Can be classified as:
  – Identifiers
  – Reserved words
  – Delimiters
  – Literals
  – Comments
Identifiers

• An identifier is the name given to a PL/SQL object, including any of the following:

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Function</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exception</td>
<td>Constant</td>
<td>Package</td>
</tr>
<tr>
<td>Record</td>
<td>PL/SQL table</td>
<td>Cursor</td>
</tr>
</tbody>
</table>

• Do not be concerned if you do not know what all of the above objects are.

• You will learn about PL/SQL objects throughout this course.
Identifiers Highlighted

Several identifiers are highlighted in the PL/SQL code shown below.

```
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;
```

Key: Variables  Packages  Procedures  Functions
Identifier Properties

Identifiers:
• Maximum 30 characters in length
• Must begin with a letter
• May include $ (dollar sign), _ (underscore), or # (hashtag)
• May not contain spaces
• Identifiers are NOT case sensitive
Recognizing PL/SQL Lexical Units

Valid and Invalid Identifiers

• Examples of valid identifiers:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First_Name</td>
<td>LastName</td>
</tr>
<tr>
<td>address_1</td>
<td></td>
</tr>
<tr>
<td>ID#</td>
<td>Total_$</td>
</tr>
<tr>
<td>primary_department_contact</td>
<td></td>
</tr>
</tbody>
</table>

• Examples of invalid identifiers:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Name</td>
<td>Contains a space</td>
</tr>
<tr>
<td>Last-Name</td>
<td>Contains invalid symbol &quot;-&quot;</td>
</tr>
<tr>
<td>1st_address_line</td>
<td>Begins with a number</td>
</tr>
<tr>
<td>Total_%</td>
<td>Contains invalid symbol &quot;%&quot;</td>
</tr>
<tr>
<td>primary_building_department_contact</td>
<td>More than 30 characters</td>
</tr>
</tbody>
</table>
Reserved Words

- Reserved words are words that have special meaning to the Oracle database.
- Reserved words cannot be used as identifiers in a PL/SQL program.
The following is a partial list of reserved words.

<table>
<thead>
<tr>
<th>ALL</th>
<th>CREATE</th>
<th>FROM</th>
<th>MODIFY</th>
<th>SELECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER</td>
<td>DATE</td>
<td>GROUP</td>
<td>NOT</td>
<td>SYNONYM</td>
</tr>
<tr>
<td>AND</td>
<td>DEFAULT</td>
<td>HAVING</td>
<td>NULL</td>
<td>SYSDATE</td>
</tr>
<tr>
<td>ANY</td>
<td>DELETE</td>
<td>IN</td>
<td>NUMBER</td>
<td>TABLE</td>
</tr>
<tr>
<td>AS</td>
<td>DESC</td>
<td>INDEX</td>
<td>OR</td>
<td>THEN</td>
</tr>
<tr>
<td>ASC</td>
<td>DISTINCT</td>
<td>INSERT</td>
<td>ORDER</td>
<td>UPDATE</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>DROP</td>
<td>INTEGER</td>
<td>RENAME</td>
<td>VALUES</td>
</tr>
<tr>
<td>CHAR</td>
<td>ELSE</td>
<td>INTO</td>
<td>ROW</td>
<td>VARCHAR2</td>
</tr>
<tr>
<td>COLUMN</td>
<td>EXISTS</td>
<td>IS</td>
<td>ROWID</td>
<td>VIEW</td>
</tr>
<tr>
<td>COMMENT</td>
<td>FOR</td>
<td>LIKE</td>
<td>ROWNUM</td>
<td>WHERE</td>
</tr>
</tbody>
</table>

Note: For more information, refer to the “PL/SQL User’s Guide and Reference.”
Recognizing PL/SQL Lexical Units

What happens when you try to use a reserved word as an identifier in a PL/SQL program?

```sql
DECLARE
  date DATE;
BEGIN
  SELECT ADD_MONTHS(SYSDATE,3) INTO date
  FROM dual;
END;
```

ORA-06550: line 4, column 37:
PL/SQL: ORA-00936: missing expression
ORA-06550: line 4, column 3:
PL/SQL: SQL Statement ignored
2. date DATE;
3. BEGIN
4. SELECT ADD_MONTHS(SYSDATE,3) INTO date
5. FROM DUAL;
6. END;

Delimiters

- Delimiters are symbols that have special meaning.
- Simple delimiters consist of one character.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>addition operator</td>
</tr>
<tr>
<td>-</td>
<td>subtraction/negation operator</td>
</tr>
<tr>
<td>*</td>
<td>multiplication operator</td>
</tr>
<tr>
<td>/</td>
<td>division operator</td>
</tr>
<tr>
<td>=</td>
<td>equality operator</td>
</tr>
<tr>
<td>'</td>
<td>character string delimiter</td>
</tr>
<tr>
<td>;</td>
<td>statement terminator</td>
</tr>
</tbody>
</table>
## Delimiters

Compound delimiters consist of two characters.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&gt;</td>
<td>inequality operator</td>
</tr>
<tr>
<td>!=</td>
<td>inequality operator</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>single-line comment indicator</td>
</tr>
<tr>
<td>/*</td>
<td>beginning comment delimiter</td>
</tr>
<tr>
<td>*/</td>
<td>ending comment delimiter</td>
</tr>
<tr>
<td>**</td>
<td>exponent</td>
</tr>
<tr>
<td>:=</td>
<td>assignment operator</td>
</tr>
</tbody>
</table>
Literals

• A literal is an explicit numeric, character string, date, or Boolean value that might be stored in a variable.

• Literals are classified as:
  – Character (also known as string literals)
  – Numeric
  – Boolean
Character Literals

Character literals:

• May include any printable character in the PL/SQL character set: letters, numerals, spaces, and symbols

• Typically defined using the VARCHAR2 data type

• Must be enclosed by character string delimiters (')

• Can be composed of zero or more characters

• Are case sensitive; therefore, PL/SQL is NOT equivalent to pl/sql
Character Literals

• The following are examples of character literals being assigned to variables.

• The literals are the characters between the single quotes (the character string delimiters) and are shown here in red text for emphasis.

```
DECLARE
    v_firstname   VARCHAR2(30) := 'John';
    v_classroom  VARCHAR2(4)  := '12C';
    v_course_id  VARCHAR2(8)  := 'CS 101';
BEGIN
    ...
```
Numeric Literals

• Literals that represent numbers are numeric literals.

• Numeric literals can be a simple value (ex. 5, -32.5, 127634, 3.141592)

• Scientific notation also may be used (ex. 2E5, meaning 2*(10 to the power of 5)).

• Typically defined using the NUMBER data type
Numeric Literals

• The following are examples of numeric literals being assigned to variables (and one constant).

• The literals are shown here in red text for emphasis.

```sql
DECLARE
    v_classroom    NUMBER(3) := 327;
    v_grade        NUMBER(3) := 95;
    v_price        NUMBER(5) := 150;
    v_salary       NUMBER(8) := 2E5;
    c_pi           CONSTANT NUMBER(7,6) := 3.141592;
BEGIN
    ...
```
Boolean Literals

• Values that are assigned to Boolean variables are Boolean literals.

• TRUE, FALSE, and NULL are the Boolean literals.

```
DECLARE
  v_new_customer  BOOLEAN := FALSE;
  v_fee_paid      BOOLEAN := TRUE;
  v_diploma       BOOLEAN := NULL;
BEGIN
  ...
```

• Note that character string delimiters are not required.
Comments

• Comments explain what a piece of code is trying to achieve.

• Well-placed comments are extremely valuable for code readability and future code maintenance.

• It is good programming practice to comment code.

• Comments are ignored by PL/SQL.

• They make no difference to how a PL/SQL block executes or the results it displays.
Syntax for Commenting Code

Two ways to indicate comments in PL/SQL

- When commenting a single line, use two dashes (--) 
- When commenting multiple lines, begin the comment with /* and end the comment with */

```
DECLARE
    -- converts monthly salary to annual salary
    v_monthly_sal NUMBER(9,2);
    v_annual_sal NUMBER(9,2);
BEGIN
    -- begin executable section
    ...
    /* Compute the annual salary based on the 
       monthly salary input from the user */
    v_annual_sal := v_monthly_sal * 12;
END;
```
Terminology

Key terms used in this lesson included:

- Lexical units
- Identifiers
- Reserved words
- Delimiters
- Literals
- Comments
Summary

In this lesson, you should have learned how to:

• List and define the different types of lexical units available in PL/SQL

• Describe identifiers and identify valid and invalid identifiers in PL/SQL

• Describe and identify reserved words, delimiters, literals, and comments in PL/SQL