Database Programming with PL/SQL

Introduction to PL/SQL

ORACLE® ACADEMY

Objectives

This lesson covers the following objectives:

- Describe PL/SQL
- Differentiate between SQL and PL/SQL
- Explain the need for PL/SQL

Purpose

PL/SQL is Oracle Corporation's standard procedural language for relational databases. To describe PL/SQL you learn its characteristics and get better prepared to describe the differences between PL/SQL and SQL.

Understanding the limitations of SQL will help you to understand why the PL/SQL language is needed. You understand how important it is not to lose a key to your house when you are trying to get into the house without the key.

PL/SQL Description

Procedural Language extension to SQL:

- Allows basic program logic and control flow to be combined with SQL statements.
- Is an Oracle proprietary programming language.
  - – It can be used only with an Oracle database or tool.

IF...THEN..
ELSE
...
END IF;
LOOP...
END LOOP;
SQL
PL/SQL

PL/SQL Description (cont.)

Procedural Language extension to SQL:

- Is a procedural language.
  - It produces a result when a series of instructions are followed.
- Is a 3GL (third-generation programming language).
  - It is a “high-level” programming language.

Structured Query Language (SQL) Description

SQL:

- Is the primary language used to access and modify data in a relational database.
- Is a nonprocedural language.
  - Also known as a "declarative language," it allows the programmer to focus on input and output rather than the program steps.
- Is a 4GL (fourth-generation-programming language).
  - A language that is closer to natural language than a programming language; query languages are generally 4GL.

Structured Query Language (SQL) Description (cont.)

SQL:

- Is a common query language for many types of databases, including Oracle.
- Has been standardized by the American National Standards Institute (ANSI).

SQL Statement

The SQL statement shown is simple and straightforward. However, if you want to alter any of the retrieved data in a conditional manner (if the data is xyz then do this to it), you come across the limitations of SQL.

```
SELECT class_id, stu_id,
  final_numeric_grade, final_letter_grade
FROM enrollments;
```

For example, how would you write an SQL statement to update the final_letter_grade data with varying letter grades for students in different classes?

Limitations of SQL

```
For class_id=1
If 66<final_numeric_grade<75 then final_letter_grade=A
If 56<final_numeric_grade<65 then final_letter_grade=B
If 46<final_numeric_grade<55 then final_letter_grade=C
If 36<final_numeric_grade<45 then final_letter_grade=D
Otherwise, final_letter_grade=F
```

CLASS_ID	STU_ID	FINAL_NUMERIC_GRADE	FINAL_LETTER_GRADE
1	101	75	
1	107	71	
1	131	65	
2	155	91	
2	114	93	

```
For class_id=2
If 91<numeric_grade<100 then  final_letter_grade=A
If 81<numeric_grade<90 then  final_letter_grade=B
If 71<numeric_grade<80 then  final_letter_grade=C
If 71<numeric_grade<80 then  final_letter_grade=D
Otherwise, final_letter_grade=F
```

Limitations of SQL (cont.)

One solution to updating the final letter grade data is shown. How many SQL statements do you need to write for class_id=1? For class_id=2? What if there were 20 classes?

```
UPDATE enrollments
SET final_letter_grade='A'
WHERE class_id=1 AND
Final_numeric_grade BETWEEN 66 and 75;

UPDATE enrollments
SET final_letter_grade='B'
WHERE class_id=1 AND
Final_numeric_grade between 56 and 65;

And so on…
```

Limitations of SQL (cont.)

One solution is to write one SQL statement for each class_id plus number_grade combination. This results in five SQL statements for class_id=1:

```
UPDATE enrollments SET final_letter_grade='A'
       WHERE class_id=1
   AND final_numeric_grade BETWEEN 66 and 75;
UPDATE enrollments SET final_letter_grade='B'
       WHERE class_id=1
   AND final_numeric_grade BETWEEN 56 and 65;
UPDATE enrollments SET final_letter_grade='C'
       WHERE class_id=1
   AND final_numeric_grade BETWEEN 46 and 55;
UPDATE enrollments SET final_letter_grade='D'
       WHERE class_id=1
   AND final_numeric_grade BETWEEN 36 and 45;
UPDATE enrollments SET final_letter_grade='F'
       WHERE class_id=1
       AND final_numeric_grade <=35;
```

Limitations of SQL (cont.)

This is a lot of statements and it does not even include the statements for the other class IDs! Similarly, there would be five statements for class_id=2.

It would be easier to write a single statement to accomplish this task. The statement would require logic, otherwise known as conditional or procedural logic.

PL/SQL extends SQL with procedural logic.

PL/SQL Extends SQL With Procedural Logic

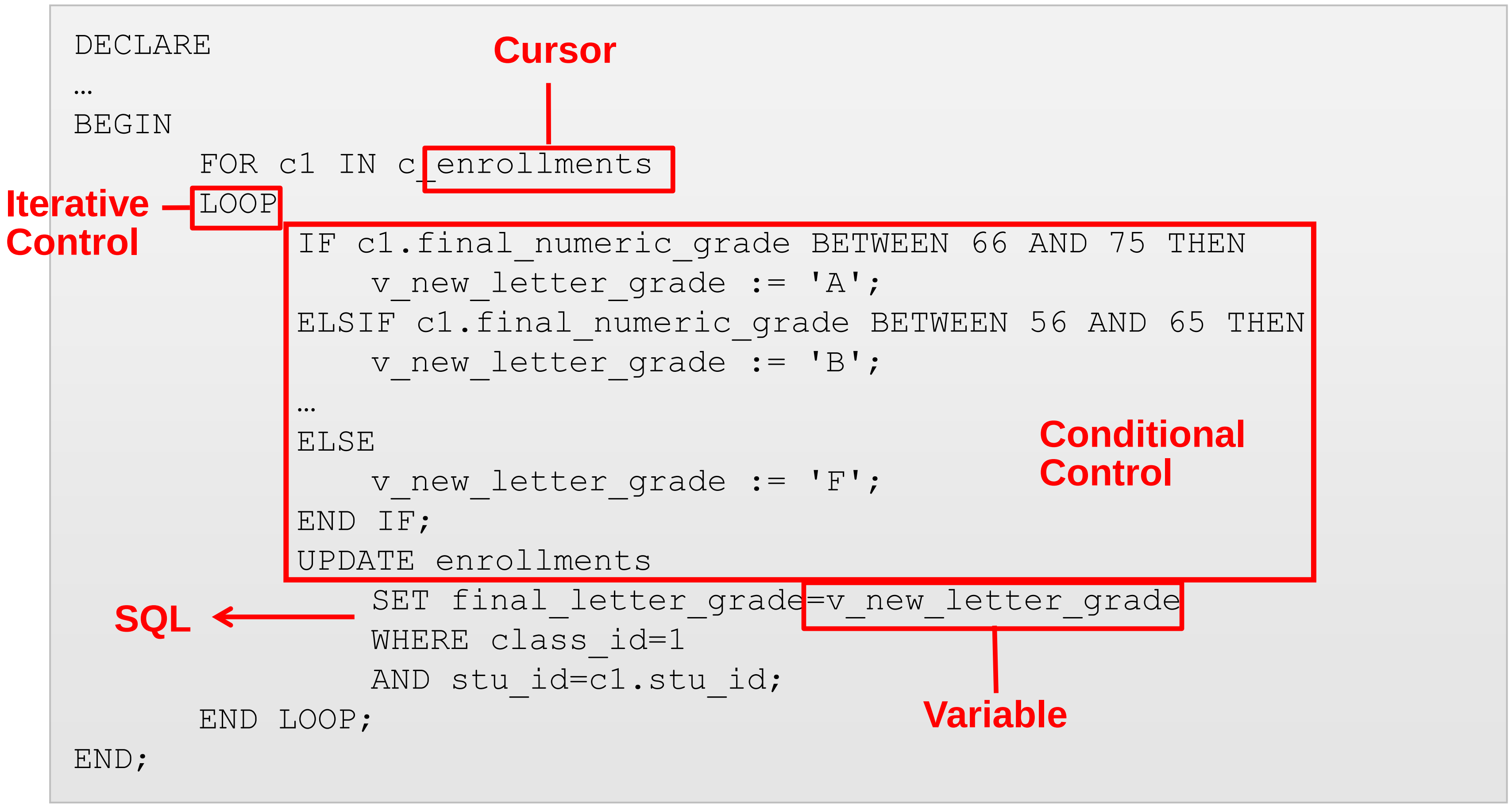
```
DECLARE
 v_new_letter_grade varchar2(1);
 CURSOR c_enrollments IS
      SELECT stu_id, final_numeric_grade FROM enrollments WHERE class_id=1;
BEGIN
 FOR c1 in c_enrollments
 LOOP
   IF c1.final_numeric_grade BETWEEN 66 and 75 THEN v_new_letter_grade := 'A';
   ELSIF c1.final_numeric_grade BETWEEN 56 AND 65 THEN v_new_letter_grade := 'B';
   ELSIF c1.final_numeric_grade BETWEEN 46 AND 55 THEN v_new_letter_grade := 'C';
   ELSIF c1.final_numeric_grade BETWEEN 36 AND 45 THEN v_new_letter_grade := 'D';
   ELSE
       v_new_letter_grade := 'F';
   END IF;
   UPDATE enrollments
     SET final_letter_grade=v_new_letter_grade WHERE class_id=1
     AND stu_id=c1.stu_id;
 END LOOP;
 COMMIT;
END;
```

Procedural Constructs

You use PL/SQL to write the procedural code, and embed SQL data-accessing statements within the PL/SQL code.

- The PL/SQL code uses variables, cursors, and conditional logic.
- PL/SQL provides procedural constructs, such as:
  - Variables, constants, and types
  - Control structures, such as conditional statements and loops
  - Reusable program units that are written once and executed many times

Procedural Constructs Highlighted



```
DECLARE
…
BEGIN
      FOR c1 IN c_enrollments
      LOOP
           IF c1.final_numeric_grade BETWEEN 66 AND 75 THEN
               v_new_letter_grade := 'A';
           ELSIF c1.final_numeric_grade BETWEEN 56 AND 65 THEN
               v_new_letter_grade := 'B';
           …
           ELSE
               v_new_letter_grade := 'F';
           END IF;
           UPDATE enrollments
               SET final_letter_grade=v_new_letter_grade
               WHERE class_id=1
               AND stu_id=c1.stu_id;
      END LOOP;
END;
```

Characteristics of PL/SQL

PL/SQL:

- Is a highly structured, readable, and accessible language.
- Is a standard and portable language for Oracle development.
- Is an embedded language and it works with SQL.
- Is a high-performance, highly integrated database language.
- Is based on the ADA language and has many similarities in syntax.

Terminology

Key terms used in this lesson included:

- PL/SQL
- Procedural Constructs

Summary

In this lesson, you should have learned how to:

- Describe PL/SQL
- Differentiate between SQL and PL/SQL
- Explain the need for PL/SQL