CSCI235 Database Systems

PL/SQL

Dr Janusz R. Getta

School of Computing and Information Technology - University of Wollongong

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

PL/SQL? What is it? Why do we need it?

```
PL/SQL is a procedural extension of SQL

PL/SQL = procedural Programming Language + SQL

We need PL/SQL to bridge a gap between a high level declarative query language and a procedural programming language

PL/SQL is a subset of a programming language Ada

PL/SQL =
```

- Data Manipulation statements of SQL +
- SELECT statement +
- variables +
- assignment statement +
- conditional control statements +
- repetition statement +
- exception handling +
- procedure and function statements + packages

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

3/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

Program structure

PL/SQL is a block-structured language

It means that its basic units such as anonymous blocks, procedures, and functions are the logical blocks

Anonymous block is persistent for only a single processing, i.e. it is not stored in a data dictionary

A named block (either procedure or function) is persistent for many processings, i.e. it can be stored in a data dictionary

Logical blocks can be nested to any level

Logical blocks consist of declarative, executable, and exception components

A declarative component consists of declarations of constants, variables, types, methods, cursors, etc, and it is optional

An executable component consists of executable code and must have at least one statement

TOP

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

Program structure

An exception component consists of executable code handling exceptions and it is optional

A sample anonymous block

```
PL/SQL
-- A sample single line comment
                        -- A keyword, beginning of declarative component
DECLARE
       Declarative
                           A sample multiline comment
                                                                               */
        component
                                                                              PL/SOL
                        -- A keyword, the beginning of executable component
BEGIN
/*
       Executable component
                                                                               */
                         -- it must include at least one statement,
NULL;
                         -- NULL; is an optional empty statement
                                                                              PL/SOL
EXCEPTION
                        -- A keyword, the beginning of exception component
       Exception component
                                                                                          */
                        -- A keyword, the end of anonymous block
END:
                        -- A forwad slash line means: execute this procedure
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

6/36

TOP

6 of 36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

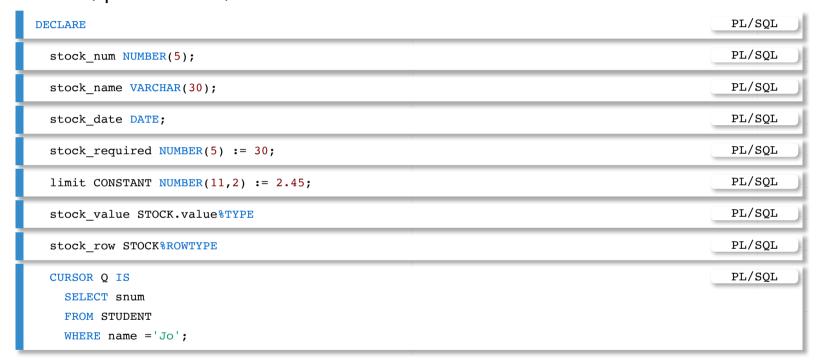
Control statements

Cursors

Exceptions

Declarative components

Declarative components contain declarations of variables, constants, cursors, procedures, and functions



TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

Executable components

Executable components include assignment statements, conditional control statements, iterative statements, procedure and function calls, SQL statements

```
student num := 910000;
                                                                                                  PL/SQL
                                                                                                  PL/SQL
SELECT name
INTO student name
FROM STUDENT
WHERE s# = student num;
                                                                                                  PL/SOL
IF (a > b)
THEN
 a := a + 1;
 c := c + 2
ELSIF (a < b)
 c := c - 2
ELSE
 b := b + 1
END IF:
                                                                                                  PL/SOL
FOR i IN 1..100 LOOP
  b := b - i
END LOOP:
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

9/36

Exception components

Exception component consists of executable statements that service the exceptional situations during execution

```
PL/SOL
EXCEPTION
WHEN NO DATA FOUND THEN
INSERT INTO AUDIT TABLE VALUES ( SYSDATE, snum )
WHEN OTHERS
  i : = i + 1
 UPDATE DEPARTMENT
 SET budget = i * budget;
END:
DECLARE
                                                                                                 PL/SOL
 too_large EXCEPTION;
BEGIN
 IF a > 100000 THEN
     RAISE too large;
 END IF:
EXCEPTION
 WHEN too large THEN
    DBMS OUTPUT.PUT LINE ('Too large!');
END;
```

TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

10/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

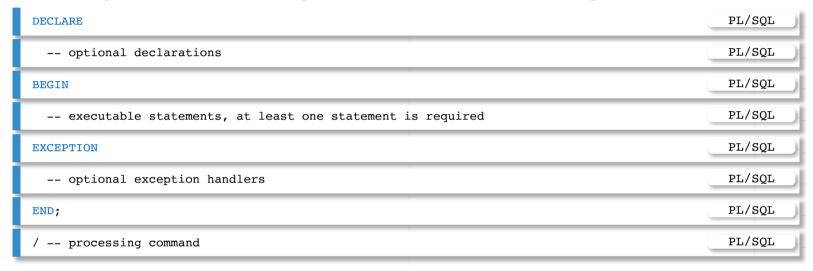
Cursors

Exceptions

TOP

Structure of anonymous block

A birds-eye view of an anonymous block is the following



A sample Hello world ! anonymous block

```
SET SERVEROUTOUT ON

BEGIN

DBMS_OUTPUT.PUT_LINE('Hello world !');
END;
/
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

12/36

A sample anonymous block

Processing SQL statements in a sample anonymous block



TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

TOP

Structure of procedure

A birds-eye view of a procedure is the following

```
PROCEDURE procedure_name ( parameters ) IS

-- optional declarations

BEGIN

-- executable statements, at least one statements is required

PL/SQL

EXCEPTION

PL/SQL

-- optional exception handlers

PL/SQL

END procedure_name;

PL/SQL
```

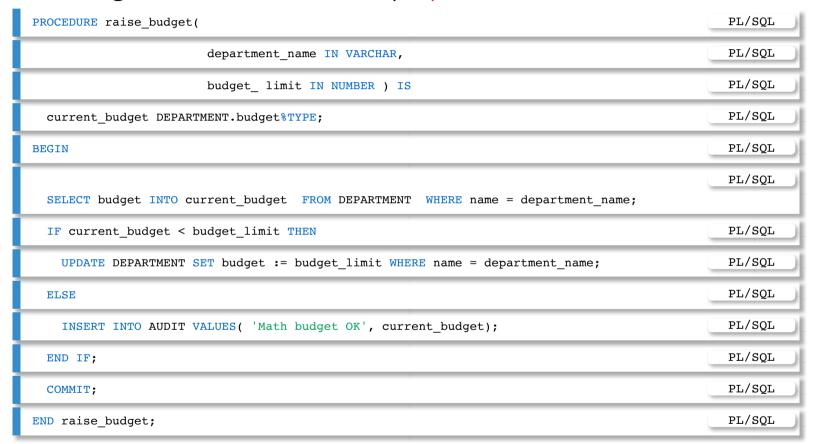
A sample hello world procedure

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

14/36

A sample procedure

Processing SQL statements in a sample procedure



TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020 15/36

Structure of function

A birds-eye view of a function is the following

```
FUNCTION function name ( parameters )
                                                                                                PL/SOL
                                                                                                PL/SQL
RETURN type-specification IS
                                                                                                PL/SQL
  -- optional declarations
                                                                                                PL/SQL
BEGIN
  -- executable statements, at least one statements is required
                                                                                                PL/SQL
EXCEPTION
                                                                                                PL/SQL
                                                                                                PL/SQL
  -- optional exception handlers
END function_name;
                                                                                                PL/SQL
```

A sample hello world function

```
FUNCTION hello_world ( hello IN VARCHAR2, world IN VARCHAR2 ) IS

RETURN VARCHAR2 IS

BEGIN

RETURN hello || ' ' || world || ' !';

END hello_world;
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

16/36

-

TOP

Structure of function

Processing SQL statements in a sample function

FUNCTION raise_budget(PL/SQL
department_name IN VARCHAR,	PL/SQL
budget_ limit IN NUMBER)	PL/SQL
RETURN NUMBER IS	PL/SQL
<pre>current_budget DEPARTMENT.budget%TYPE;</pre>	PL/SQL
BEGIN	PL/SQL
SELECT budget INTO current_budget FROM DEPARTMENT WHERE name = department_name;	PL/SQL
IF current_budget < budget_limit THEN	PL/SQL
<pre>UPDATE DEPARTMENT SET budget = budget_limit WHERE name = department_name;</pre>	PL/SQL
RETURN budget_limit;	PL/SQL
ELSE	PL/SQL
<pre>INSERT INTO AUDIT VALUES('Math budget OK', current_budget);</pre>	PL/SQL
RETURN current_budget;	PL/SQL
END IF;	PL/SQL
COMMIT;	PL/SQL
END raise_budget; Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020	PL/SQL 17/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

Data types

Some of the predefined data types in PL/SQL

```
INTEGER, DECIMAL, NUMBER, CHAR, DATE, VARCHAR, VARCHAR2, LONG,
BOOLEAN, ROWID, EXCEPTION
```

Sample implicit type declarations

```
DECLARE
                                                                              PL/SQL
 student no STUDENT.snum%TYPE;
 student name STUDENT.name%TYPE;
 student row STUDENT%ROWTYPE;
                                                                              PL/SQL
BEGIN
                                                                              PL/SQL
 student no := 1234567;
 SELECT name FROM STUDENT INTO student name WHERE snum = student no;
                                                                              PL/SQL
 student row.snum := 1234567;
                                                                              PL/SQL
 student row.name := 'James';
 student rec.dob := TO DATE('01-DEC-1994', 'DD-MON-YYYY');
 INSERT INTO STUDENT VALUES (student row.snum, student row.name, student row.dob);
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

19/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

TOP

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

Operators

Arithmetic operators

+, -, *, /, **
PL/SQL

Relational operators

<, >, >=, <=, =, !=, <>, ~=
PL/SQL

Comparison operators

LIKE, BETWEEN, IN, IS NULL, =, !=, <>, ~=

Boolean operators

AND, OR, NOT

String operator

| | PL/SQL

Operator precedence

(**),(unary +,-),(*,/),(+,-,||),(comparison),(NOT),(AND),(OR)

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

21/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

Conditional control statements

A birds-eye view of conditional control statements is the following

```
IF condition THEN
                                                                                                   PL/SQL
  statement;
ELSE
  statement;
END IF:
                                                                                                   PL/SQL
IF condition THEN
statement;
ELSIF condition THEN
  statement;
ELSIF condition THEN
  statement;
ELSE
  statement;
END IF;
```

TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

23/36

Iterative control statements

A birds-eye view of iterative control statements is the following

```
PL/SOL
LOOP
  statement;
  IF condition THEN EXIT;
    statement;
 END IF;
  statement;
END LOOP;
                                                                                                   PL/SQL
 FOR variable IN scope
LOOP
  statement;
END LOOP;
FOR variable IN REVERSE scope
LOOP
  statement;
END LOOP;
```

TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

Iterative control statements

A birds-eye view of iterative control statements is the following

```
WHILE (condition)

LOOP
statement;
...
END LOOP;

LOOP
statement;
...
EXIT WHEN condition;
statement;
...
END LOOP;
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

25/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

Cursors

What happens when **SELECT** statement returns more than one row?

```
DECLARE

student_no STUDENT.snum%TYPE;

BEGIN

SELECT snum

INTO student_no

FROM STUDENT

WHERE name = 'Pam';

...

ERROR at line 1:

ORA-06503: PL/SQL: error 0 - Unhandled exception ORA-01427: single-row subquery returns more than one row which was raised in a statement ending at line 6
```

A variable student_no cannot be used to store several rows retrieved
from a relational table

A solution is to process the rows in a row by row mode

A cursor is a construction that allows for processing the rows retrieved from the relational tables in a row by row mode

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

27/36

27 of 36

Cursors

Explicit declaration and processing of a cursor

```
DECLARE
                                                                               PL/SOL
  student no STUDENT.snum%TYPE;
  CURSOR Q IS
                                                                               PL/SQL
    SELECT snum
    FROM STUDENT
    WHERE name = 'Pam';
BEGIN
                                                                               PL/SQL
 OPEN Q;
 LOOP
   FETCH Q INTO student_no;
    IF Q%NOTFOUND THEN
      EXIT;
    END IF;
    INSERT INTO PAM VALUES (student no)
  END LOOP;
 CLOSE Q;
 COMMIT;
END;
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

28/36

Implicit cursor processing

Implicitly declaration and processing of a cursor

A cursor is implicitly declared

A cursor is implicitly opened

A row is implicitly fetched

End of table condition is implicitly checked

A cursor is implicitly closed

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

29/36

Cursor attributes

A cursor attribute determines a state of a cursor

A cursor attribute **%NOTFOUND** evaluates to true if the last **FETCH** failed because no more rows were available

A cursor attribute %FOUND evaluates to true if the last FETCH succeeded

A cursor attribute %ROWCOUNT evaluates to the total number of rows FETCHed so far

A cursor attribute %ISOPEN evaluates to true if a cursor is opened

You can find more information about cursor attributes here

TOP

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

Cursor attributes

A sample testing of cursor attributes

```
DECLARE
                                                                              PL/SOL
 student no STUDENT.snum%TYPE;
 CURSOR Q IS
    SELECT snum FROM STUDENT WHERE name = 'Pam';
BEGIN
 OPEN Q;
 LOOP
    FETCH Q INTO student_no;
    IF Q%NOTFOUND THEN
      EXIT
    END IF:
    INSERT INTO PAM VALUES(student no);
 END LOOP;
 IF QROWCOUNT = 0 THEN
                                                                              PL/SQL
    INSERT INTO MESSAGES VALUES ('NO ROWS PROCESSED');
 END IF;
 CLOSE Q;
 COMMIT;
END;
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

31/36

Outline

PL/SQL? What is it? Why do we need it?

Program structure

Declarative, Executable, Exception components

Structures of anonymous blocks, procedures, and functions

Data types, implicit type declarations

Operators

Control statements

Cursors

Exceptions

TOP

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

Exceptions

TOP

An exception is an internally defined or user defined error condition, e.g. divide by zero, no rows selected by SELECT statement with INTO clause, failure of FETCH statement, use of a cursor which has not been opened yet, etc.

A typical exception handling

```
DECLARE
    error_number NUMBER(5);
    error_message VARCHAR(200);
...

EXCEPTION
    WHEN OTHERS THEN
        error_number = SQLCODE;
        error_message = SQLERRM;
        DBMS_OUTPUT.PUT_LINE(error_number ||'-'|| error_message);
        INSERT INTO ERRORS( error_number, error_message);
        COMMIT;
END;
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

33/36

Exceptions

Handling empty an answer from **SELECT** statement

```
DECLARE

student_name STUDENT.name%TYPE;

BEGIN

SELECT name

INTO student_name

FROM STUDENT

WHERE snum = 1234567;

...

EXCEPTION

WHEN NO_DATA_FOUND THEN

INSERT INTO MESSAGES VALUES( 'Student not found');

COMMIT;

END;
```

Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020

34/36

Exceptions

An exception NO_DATA_FOUND is raised when SELECT statement returns no rows

An exception TOO_MANY_ROWS is raised when SELECT statement returns more than one row

An exception INVALID_CURSOR is raised when PL/SQL call specifies an invalid cursor, e.g. closing an unopened cursor

An exception OTHERS is raised when any other exception, not explicitly named happens

You can find a complete list of PL/SQL exceptions here

References

Database PL/SQL Language Reference

T. Connoly, C. Begg, Database Systems, A Practical Approach to Design, Implementation, and Management, Chapter 8 Advanced SQL, Pearson Education Ltd, 2015

TOP Created by Janusz R. Getta, CSCI235 Database Systems, Spring 2020