PL/SQL

wprowadzenie

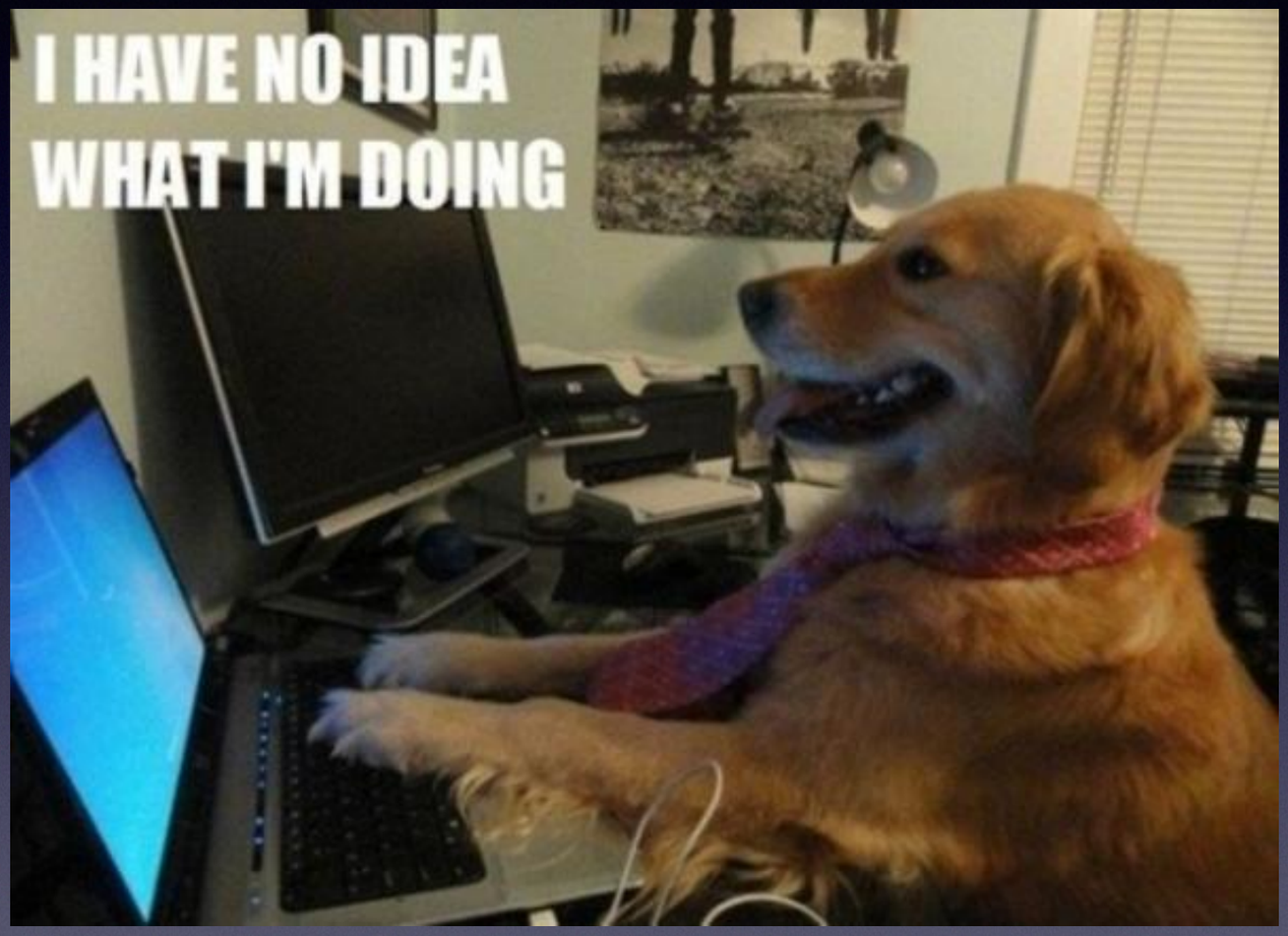
Co to jest PL/SQL?

(pgPL/SQL?)

Zalety

- Struktura blokowa
- Język proceduralny
- Wydajność
- Obsługa błędów

Wady



Łatwo o pomyłkę która będzie kosztowała dużo czasu

Blok PL/SQL

- Sekcja deklaracji (opcjonalna)
- Sekcja wykonywana
- Sekcja wyjątków/błędów (opcjonalna)

Blok PL/SQL

```
--komentarz
DECLARE
      Deklaracja zmiennych
BEGIN
      Wykonanie
EXCEPTION
      Błędy
END;
/*komentarz wieloliniowy blah
blah blah*/
```

Blok pgPL/SQL

```
DO $$
--komentarz
DECLARE
      Deklaracja zmiennych
BEGIN
      Wykonanie
EXCEPTION
      Błędy
END;
/*komentarz wieloliniowy blah
blah blah*/
$$ LANGUAGE plpgsql;
```

Zmienne

Składnia deklaracji PL/SQL

```
nazwa_zmiennej typ [NOT NULL := wartość_domyślna ];
```

- nazwa_zmiennej
- typ
- NOT NULL (opcjonalnie)
- wartość_domyślna (opcjonalnie)

Zmienne

Składnia deklaracji PL/SQL

```
DECLARE
city_code city.city_id%TYPE;
city_name city.city%TYPE NOT NULL := “Warsaw”;
country character varying(50);
```

Stałe

Składnia deklaracji

```
nazwa_stałej CONSTANT typ:= wartość;
```

- nazwa
- typ
- wartość

Instrukcje warunkowe

```
IF condition 1
THEN
 statement 1;
 statement 2;
ELSIF condtion2 THEN
 statement 3;
ELSE
 statement 4;
END IF
```

Instrukcje iteracyjne

```
LOOP
   statements;
   EXIT;
   {or EXIT WHEN condition;}
END LOOP;
```

```
WHILE <condition>
 LOOP statements;
END LOOP;
```

```
FOR counter IN val1..val2
  LOOP statements;
END LOOP;
```

Demo

Co to jest procedura?

(Czym różni się od funkcji?)

Procedura

```
CREATE [OR REPLACE] PROCEDURE proc_name [list of parameters]
IS
   Declaration section
BEGIN
   Execution section
EXCEPTION
  Exception section
END;
```

Funkcja

```
CREATE [OR REPLACE] FUNCTION function_name [parameters]
RETURN return_datatype;
IS
  Declaration_section
BEGIN
  Execution_section
  Return return_variable;
EXCEPTION
  exception section
Return return_variable;
END;
```

Funkcja

Składnia deklaracji pgPL/SQL

```
CREATE FUNCTION function_name(p1 type, p2 type)
 RETURNS type AS
BEGIN
 -- logic
EXCEPTION
 -- error handling
END;
LANGUAGE language_name;
```

Demo

Rekordy

Składnia deklaracji PL/SQL

```
TYPE record_type_name IS RECORD
(first_col_name column_datatype,
second_col_name column_datatype,
…);
specific_record city%ROWTYPE;
```

Rekordy

Składnia deklaracji pgPL/SQL

```
record_type_name RECORD;
specific_record city%ROWTYPE;
```

Demo

Co to jest kursor?

Kursory domyślne

- Nie wymagają deklaracji
- Instrukcje SELECT, UPDATE, DELETE i INSERT
- Mogą zwracać tylko jeden wiersz (lub nic)

Kursory domyślne

```
DECLARE  var_rows number(5);
BEGIN
  UPDATE employee
  SET salary = salary + 1000;
  IF SQL%NOTFOUND THEN
    dbms_output.put_line('None of the salaries where updated');
  ELSIF SQL%FOUND THEN
    var_rows := SQL%ROWCOUNT;
    dbms_output.put_line('Salaries for ' || var_rows || 'employees are updated');
  END IF;
END;
```

Kursory definiowane

- Wymagają deklaracji
- Instrukcje SELECT
- Mogą zwracać więcej niż jeden wiersz

Kursory definiowane

```
DECLARE
   CURSOR emp_cur IS
   SELECT first_name, last_name, salary FROM emp_tbl;
   emp_rec emp_cur%rowtype;
BEGIN
OPEN sales_cur;
LOOP
     FETCH emp_cur INTO emp_rec;
     EXIT WHEN emp_cur%NOTFOUND;
     dbms_output.put_line(emp_cur.first_name || ' ' ||emp_cur.last_name
     || ' ' ||emp_cur.salary);
END LOOP;
CLOSE sales_cur;
END;
```

Demo

One more thing…

Właśnie zaczął się weekend 😃