

Procedural Language/Structured Query Language (PL/SQL)

Các tính năng chính của PL/SQL

- Khối lệnh PL/SQL
- PL/SQL Input và Output
- Biến và hằng số trong PL/SQL
- Cấu trúc điều khiển trong PL/SQL
- Quản lý lỗi trong PL/SQL
- **Trừu tượng dữ liệu PL/SQL (data abstraction)**
- Chương trình con PL/SQL (Subprogram)
- PL/SQL Packages

Trừu tượng dữ liệu PL/SQL

- Record
- Thuộc tính %TYPE
- Thuộc tính %ROWTYPE
- Tập hợp (Collection)
- **Con trỏ (Cursor)**

Contrỏ (Cursor)

- Cursor là một con trỏ tới một vùng nhớ SQL lưu trữ thông tin về việc xử lý một câu lệnh SELECT hoặc DML.
- PL/SQL sử dụng con trỏ tường minh (explicit cursor) và con trỏ tiềm ẩn (implicit cursor).
- Implicit Cursors
 - PL/SQL tạo một implicit cursor mỗi khi chạy một câu lệnh SELECT hoặc DML.
- Explicit Cursors
 - Ta phải khai báo và định nghĩa một explicit cursor, đặt tên và gắn nó với một câu query.

Quản lý con trỏ (Cursor) trong PL/SQL

- Con trỏ tiềm ẩn (Implicit cursor hoặc SQL cursor)
- Con trỏ tường minh (Explicit cursor)

SQL Cursors (Implicit)

- Con trỏ tiềm ẩn được quản lý một cách tự động bởi PL/SQL.
- Ta không cần phải viết code để quản lý những con trỏ này. Tuy nhiên có thể truy vết thông tin thông qua các thuộc tính của nó.

SQL Cursors (Implicit)

Thuộc tính

- %FOUND: câu lệnh DML có thay đổi row nào không?
- %ISOPEN: luôn luôn là FALSE
- %NOTFOUND: câu lệnh DML có thất bại trong việc thay đổi row không?
- %ROWCOUNT: bao nhiêu row bị tác động?

SQL Cursors (Implicit)

thuộc tính %FOUND

- %FOUND trả về TRUE nếu câu lệnh INSERT, UPDATE, hay DELETE tác động một hay nhiều row.
- Hoặc câu lệnh SELECT INTO trả về một hay nhiều row.
- Ngược lại, %FOUND có giá trị là FALSE.

```
DECLARE
    dept_no NUMBER(4) := 270;
BEGIN
    DELETE FROM departments WHERE department_id = dept_no;
    IF SQL%FOUND THEN -- delete succeeded
        INSERT INTO departments VALUES (270, 'Personnel', 200, 1700);
    END IF;
END;
```

SQL Cursors (Implicit)

thuộc tính %ROWCOUNT

- %ROWCOUNT trả về số dòng (row) bị tác động bởi câu lệnh INSERT, UPDATE, DELETE hoặc SELECT INTO.

```
DECLARE
    mgr_no NUMBER(6) := 122;
BEGIN
    DELETE FROM employees WHERE manager_id = mgr_no;
    DBMS_OUTPUT.PUT_LINE
        ('Number of employees deleted: ' || TO_CHAR(SQL%ROWCOUNT));
END;
```

Chú ý khi sử dụng SQL Cursor (Implicit)

- Giá trị của thuộc tính con trỏ luôn là giá trị của câu lệnh SQL được thực thi gần nhất, bất chấp câu lệnh đó nằm ở đâu. Nó có thể nằm ở phạm vi khác nhau (ví dụ, trong một sub-block)
- Để lưu trữ giá trị của các thuộc tính để sau này sử dụng, nên gán nó vào một biến cục bộ.
- Thuộc tính %NOTFOUND không hữu ích khi kết hợp với câu lệnh SELECT INTO.

Con trỏ tường minh (Explicit Cursor)

- Khi ta cần điều khiển việc xử lý truy vấn, ta có thể khai báo tường minh một con trỏ trong phần khai báo của khối lệnh PL/SQL, chương trình con (subprogram), hoặc package
- Có thể sử dụng ba câu lệnh sau để điều khiển con trỏ: OPEN, FETCH, và CLOSE.

Explicit Cursors

khai báo một con trỏ

```
DECLARE
    my_emp_id NUMBER(6); -- variable for employee_id
    my_job_id VARCHAR2(10); -- variable for job_id
    my_sal NUMBER(8,2); -- variable for salary

    CURSOR c1 IS
        SELECT employee_id, job_id, salary FROM employees
        WHERE salary > 2000;
    CURSOR c2 IS
        SELECT * FROM departments WHERE department_id = 110;
DECLARE
    CURSOR e1 (high NUMBER) IS
        SELECT * FROM departments WHERE department_id < high;
```

Explicit Cursor mở một con trỏ

```
DECLARE
  CURSOR c1 IS
    SELECT employee_id, last_name, job_id, salary
    FROM employees
    WHERE salary > 2000;
BEGIN
  OPEN c1;
  ....
END;
```

Explicit Cursors sử dụng FETCH

```
DECLARE
  v_jobid      employees.job_id%TYPE; -- variable for job_id
  v_lastname   employees.last_name%TYPE; -- variable for last_name

  CURSOR c1 IS SELECT last_name, job_id FROM employees
  WHERE job_id LIKE '%CLERK';
BEGIN
  OPEN c1; -- open the cursor before fetching
  LOOP
    -- Fetches 2 columns into variables
    FETCH c1 INTO v_lastname, v_jobid;
    EXIT WHEN c1%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE( RPAD(v_lastname, 25, ' ') || v_jobid );
  END LOOP;
  CLOSE c1;
END;
```

Explicit Cursors sử dụng FETCH

```
DECLARE
  v_employees employees%ROWTYPE; -- record variable for row
  CURSOR c2 IS SELECT * FROM employees
  WHERE job_id LIKE '%CLERK';
BEGIN
  OPEN c2;
  LOOP
    -- Fetches entire row into the v_employees record
    FETCH c2 INTO v_employees;
    EXIT WHEN c2%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE
      ( v_employees.last_name || v_employees.job_id );
  END LOOP;
  CLOSE c2;
END;
```

Explicit Cursors đóng một con trỏ

- Câu lệnh CLOSE vô hiệu hóa con trỏ và tập kết quả trở thành không xác định
- Một khi con trỏ được đóng lại, ta có thể mở lại nó.
- Bất cứ một thao tác nào trên con trỏ đã bị đóng đều gây nên một exception INVALID_CURSOR;

Attributes of Explicit Cursors

- Thuộc tính %FOUND.
- Thuộc tính %ISOPEN.
- Thuộc tính %NOTFOUND.
- Thuộc tính %ROWCOUNT.

Explicit Cursors thuộc tính %FOUND

```
DECLARE
  CURSOR c1 IS SELECT last_name, salary FROM employees WHERE
  ROWNUM < 11;
  my_ename   employees.last_name%TYPE;
  my_salary  employees.salary%TYPE;
BEGIN
  OPEN c1;
  LOOP
    FETCH c1 INTO my_ename, my_salary;
    IF c1%FOUND THEN -- fetch succeeded
      DBMS_OUTPUT.PUT_LINE
        ('Name = ' || my_ename || ', salary = ' || my_salary);
    ELSE -- fetch failed, so exit loop
      EXIT;
    END IF;
  END LOOP;
END;
```

Explicit Cursors thuộc tính %ISOPEN

```
DECLARE
CURSOR c1 IS
SELECT last_name, salary FROM employees WHERE ROWNUM < 11;

the_name employees.last_name%TYPE;
the_salary employees.salary%TYPE;
BEGIN
IF c1%ISOPEN = FALSE THEN -- cursor was not already open
OPEN c1;
END IF;
FETCH c1 INTO the_name, the_salary;
CLOSE c1;
END;
```

Explicit Cursors thuộc tính %NOTFOUND

```
DECLARE
CURSOR c1 IS SELECT last_name, salary
FROM employees
WHERE ROWNUM < 11;
my_name employees.last_name%TYPE;
my_salary employees.salary%TYPE;
BEGIN
OPEN c1;
LOOP
FETCH c1 INTO my_name, my_salary;
IF c1%NOTFOUND THEN -- fetch failed, so exit loop
-- Another form of this test is
-- "EXIT WHEN c1%NOTFOUND OR c1%NOTFOUND IS NULL;"
EXIT;
ELSE -- fetch succeeded
DBMS_OUTPUT.PUT_LINE ('Name = ' || my_name || ', salary = ' || my_salary);
END IF;
END LOOP;
END;
```

Explicit Cursors thuộc tính %ROWCOUNT

```
DECLARE
CURSOR c1 IS SELECT last_name FROM employees WHERE ROWNUM < 11;
name employees.last_name%TYPE;
BEGIN
OPEN c1;
LOOP
FETCH c1 INTO name;
EXIT WHEN c1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(c1%ROWCOUNT || ' ' || name);
IF c1%ROWCOUNT = 5 THEN
DBMS_OUTPUT.PUT_LINE('--- Fetched 5th record ---');
END IF;
END LOOP;
CLOSE c1;
END;
```

Truyền đối số tới Explicit Cursor

```
DECLARE
emp_job employees.job_id%TYPE := 'ST_CLERK';
emp_salary salary employees.salary%TYPE := 3000;
my_record employees%ROWTYPE;
CURSOR c1 (job VARCHAR2, max_wage NUMBER) IS
SELECT * FROM employees WHERE job_id = job AND salary > max_wage;
BEGIN
OPEN c1(emp_job, emp_salary);
LOOP
FETCH c1 INTO my_record;
EXIT WHEN c1%NOTFOUND;
-- process data record
DBMS_OUTPUT.PUT_LINE ('Name = ' || my_record.last_name || ', salary = '
|| my_record.salary || ', Job Id = ' || my_record.job_id);
END LOOP;
END;
```

Cursor FOR LOOP

- SQL Cursor FOR LOOP
- Explicit Cursor FOR LOOP

SQL Cursor FOR LOOP

```
BEGIN
FOR item IN
( SELECT last_name, job_id
FROM employees
WHERE job_id LIKE '%CLERK%'
AND manager_id > 120 )
LOOP
DBMS_OUTPUT.PUT_LINE
('Name = ' || item.last_name || ', Job = ' || item.job_id);
END LOOP;
END;
```

Explicit Cursor FOR LOOP

```
DECLARE
  CURSOR c1 IS SELECT last_name, job_id FROM employees
    WHERE job_id LIKE '%CLERK%' AND manager_id > 120;
BEGIN
  FOR item IN c1
  LOOP
    DBMS_OUTPUT.PUT_LINE
      ('Name = ' || item.last_name || ', Job = ' || item.job_id);
  END LOOP;
END;
```

Explicit Cursor FOR LOOP

```
DECLARE
  CURSOR c1 (job VARCHAR2, max_wage NUMBER) IS
    SELECT * FROM employees WHERE job_id = job AND salary > max_wage;
BEGIN
  FOR person IN c1('CLERK', 3000)
  LOOP
    -- process data record
    DBMS_OUTPUT.PUT_LINE('Name = ' || person.last_name || ', salary = ' ||
      person.salary || ', Job Id = ' || person.job_id );
  END LOOP;
END;
```

Định nghĩa alias cho giá trị biểu thức trong Cursor FOR Loop

```
BEGIN
  FOR item IN
    ( SELECT first_name || ' ' || last_name AS full_name, salary * 10 AS dream_salary
      FROM employees WHERE ROWNUM <= 5 )
  LOOP
    DBMS_OUTPUT.PUT_LINE
      (item.full_name || ' dreams of making ' || item.dream_salary);
  END LOOP;
END;
```

Sử dụng Explicit cursor với Tập hợp

```
declare
  l_rowids sys.odcidridlist;
  l_usernames sys.odcivarchar2list;
  cursor c is select rowid, username from t;
  l_limit number := 100;
begin
  open c;
  loop
    fetch c bulk collect into l_rowids, l_usernames limit l_limit;
    exit when (l_rowids.count = 0);

    for i in 1 .. l_rowids.count
    loop
      process( l_usernames(i) );
    end loop;

    forall i in 1 .. l_rowids.count
      update t set username = l_usernames(i) where rowid = l_rowids(i);
  end loop;
  close c;
end;
```

Choosing the Right Way to Query

Here are some guidelines to help you decide which technique to use:

When fetching a single row, use SELECT-INTO or EXECUTE IMMEDIATE-INTO (if your query is dynamic). Do not use an explicit cursor or a cursor FOR loop.

When fetching all the rows from a query, use a cursor FOR loop unless the body of the loop executes one or more DML statements (INSERT, UPDATE, DELETE, or MERGE). In such a case, you will want to switch to BULK COLLECT and FORALL.

Use an explicit cursor when you need to fetch with BULK COLLECT, but limit the number of rows returned with each fetch.

Use an explicit cursor when you are fetching multiple rows but might conditionally exit before all rows are fetched.

Use EXECUTE IMMEDIATE to query data only when you cannot fully construct the SELECT statement while writing your code.