PL/SQL: Verwendung von Cursor

Stephan Karrer

Cursor-Konzept

- Ein Cursor verkörpert die private SQL-Area im Speicher des Benutzers, die für SELECT- und DML-Anweisungen angelegt wird
  - impliziter Cursor:
    hat keinen Namen, nur vorgefertigter Zugriff via Cursor-Attribute möglich
  - expliziter Cursor:
    wird deklariert und programmtechnisch genutzt, meist um die Ergebnismenge einer SELECT-Anweisung zu bearbeiten

Cursor-Attribute

- %FOUND : Hat eine DML-Anweisung Zeilen getroffen?
- %ISOPEN : Ist der Cursor geöffnet? (stets FALSE für select-Cursor)
- %NOTFOUND : Wurden keine Zeilen getroffen?
- %ROWCOUNT : Wieviele Zeilen wurden getroffen?

Nutzung der Attribute bei implizitem Cursor

```
CREATE TABLE dept_temp AS SELECT * FROM departments;


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

Expliziten Cursor nutzen

```
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 REGEXP_LIKE (job_id, 'S[HT]_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;
        -- COMMIT ?
    CLOSE c1;
END;
```

Expliziten Cursor mit Records nutzen

```
DECLARE
    v_employees employees%ROWTYPE; -- record variable for row
    CURSOR c2 is SELECT * FROM employees
        WHERE REGEXP_LIKE (job_id, '[ACADFIMKSA]_M[ANGR]');
BEGIN
    OPEN c2; -- open the cursor before fetching
    LOOP     -- Fetches entire row into the v_employees record
        FETCH c2 INTO v_employees;
        EXIT WHEN c2%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(
            RPAD(v_employees.last_name, 25, ' ') ||
                                        v_employees.job_id );
    END LOOP;
    CLOSE c2;
END;
```

Explizite Cursor-For-Schleifen

```
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;
```

Cursor-For-Schleife auf Basis einer Unterabfrage

```
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;
```

Aliase für Ausdrücke

```
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;
```

Parametrisierter Cursor

```
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;
```

Verwendung von Unterabfragen

```
DECLARE
    CURSOR c1 IS
        SELECT t1.department_id, department_name, staff
          FROM departments t1,
           ( SELECT department_id, COUNT(*) as staff
              FROM employees GROUP BY department_id) t2
              WHERE t1.department_id = t2.department_id
                    AND staff >= 5;

BEGIN
    FOR dept IN c1
    LOOP
        DBMS_OUTPUT.PUT_LINE ('Department = '
        || dept.department_name || ', staff = ' || dept.staff);
END LOOP;
END;
```

Datenkonsistenz bei der Cursor-Verarbeitung

```
DECLARE
    my_emp_id NUMBER(6);
    my_job_id VARCHAR2(10);
    my_sal NUMBER(8,2);
    CURSOR c1 IS SELECT employee_id, job_id, salary
                    FROM employees FOR UPDATE;
BEGIN
    OPEN c1;
    LOOP
        FETCH c1 INTO my_emp_id, my_job_id, my_sal;
        IF my_job_id = 'SA_REP' THEN
            UPDATE employees SET salary = salary * 1.02
                WHERE CURRENT OF c1;
        END IF;
        EXIT WHEN c1%NOTFOUND;
    END LOOP;
END;
```