Курс «Базы данных»

Тема. Программирование на языке PL/SQL. Часть 3

Барабанщиков
Игорь Витальевич

План лекции

1. Явные курсоры
2. Цикл FOR с курсором
3. Создание хранимых процедур
4. Вызов хранимых процедур
5. Сопровождение хранимых процедур

Явные курсоры

- Для выполнения команд SQL сервер Oracle использует рабочие области памяти, которые называют ***частная область SQL***.
- Программист может создавать ***явные курсоры***.
- В этом случае можно обращаться к частной области SQL по имени и обращаться к находящейся в ней информации.

Типы курсоров

Тип курсора	Описание
Неявный курсор	PL/SQL создает их НЕЯВНО для всех команд DML и запросов SELECT, возвращающих только одну строку.
Явный курсор	Создаются программистом ЯВНО для запросов, возвращающих более одной строки. Программист присваивает им имя и управляет с помощью специальных команд в исполняемом разделе блока.

Активный набор

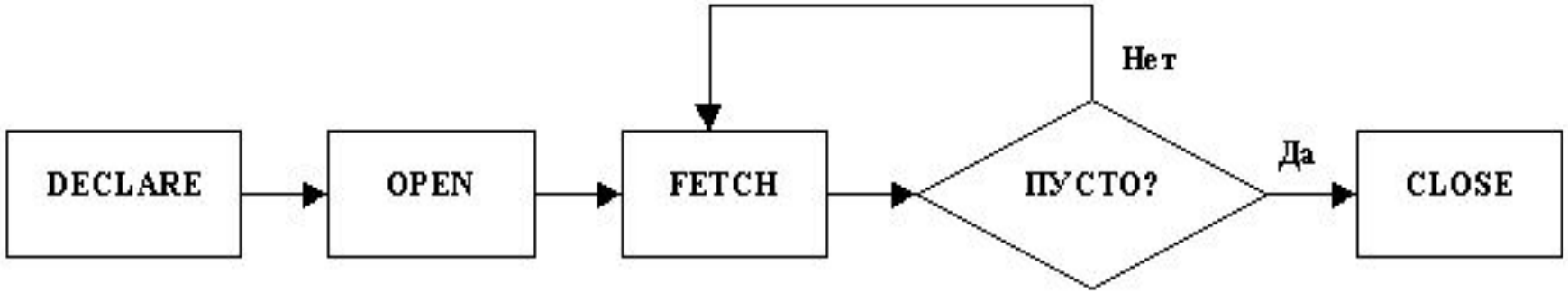
- Явные курсоры используются для ***индивидуальной*** обработки каждой строки, возвращаемой многострочной командой SELECT.
- Набор строк, возвращаемых многострочным запросом, называется ***активным набором***.
- Курсор указывает на ***текущую строку*** в активном наборе.

2111	ООО "Памира"	50000,00	103
2102	ТОО "Стройсервис"	65000,00	101
2130	Стройтрест 12	50000,00	105
2123	Стройтрест 13	40000,00	102
2107	Фирма "Барта"	20000,00	110

Текущая строка

Управление явным курсором

1. **Объявление курсора** – курсору присваивается имя и определяется структура запроса.
2. **Открытие курсора** – выполняется запрос, строки из активного набора доступны для выборки.
3. **Выборка данных** – проверка наличия и извлечение текущей строки.
4. **Закрытие курсора** – освобождается область памяти



Пример работы с курсором

```
DECLARE
   CURSOR c_emp  IS    -- 1: описание явного курсора
     SELECT  last_name  FROM  emp  WHERE  dept=10;
   v_name   emp.name%TYPE;
BEGIN
   OPEN  c_emp;          -- 2: открытие курсора
   LOOP
      FETCH  c_emp  INTO  v_name;       -- 3: извлечение
    строки
       dbms_output.put_line(v_name);
      EXIT  WHEN c_emp%NOTFOUND;  -- 3: проверка
   END  LOOP;
   CLOSE  c_emp;          -- 4: закрытие курсора
END;
```

Записи и курсоры

- Строки активного набора удобно обрабатывать, выбирая значения в переменную PL/SQL типа **RECORD**.
- Можно определить запись с использованием структуры строки таблицы.
- Это удобно для обработки строк активного набора, так как можно делать выборку прямо в запись.
- При этом значения строки загружаются в соответствующие поля записи.

Пример работы с записью

```
DECLARE
  CURSOR c_emp IS SELECT * FROM emp;
  v_emp  emp%ROWTYPE;
BEGIN
  OPEN c_emp;
  LOOP
    FETCH c_emp INTO v_emp;
    dbms_output.put_line(v_emp.name);
    EXIT WHEN c_emp%NOTFOUND;
  END LOOP;
  CLOSE c_emp;
END;
```

Циклы FOR с курсором

- Сокращенная форма кодирования операций с явными курсорами.
- Неявное открытие, выборки и закрытие.
- Запись объявляется неявно.

```
FOR имя_записи IN имя_курсора LOOP
    команда1;
    команда2;
    . . .
END LOOP;
```

Пример цикл FOR с курсором

```
DECLARE
   CURSOR c_emp  IS  SELECT  *  FROM emp;
BEGIN
   FOR  v_rec  IN  c_emp  LOOP
      dbms_output.put_line(v_rec.name);
   END  LOOP;
END;
```

Процедуры и функции

- **Это именованные блоки PL/SQL**
- Их называют подпрограммы PL/SQL
- **Они компилируются и сохраняются в БД**
- Их можно повторно использовать, вызывая по имени.
- **Имеют блочную структуру**, похожую на структуру анонимного блока:
  - декларативная секция (без DECLARE)
  - исполняемая секция
  - секция обработки исключений

Анонимные блоки и подпрограммы

Анонимные блоки	Подпрограммы
Неименованные блоки PL/SQL	Именованные блоки PL/SQL
Каждый раз компилируются	Компилируются только один раз
Не хранятся в базе данных	Хранятся в базе данных
Не могут быть вызваны из других приложений	Имеют имена и поэтому могут быть вызваны из других приложений
Не возвращают значений	Функции возвращают значение
Не могут принимать параметры	Могут принимать параметры

Синтаксис процедуры

```
CREATE [OR REPLACE] PROCEDURE имя_проц
 [ (аргумент1 [тип_параметра1]
  тип_данных1,
 [  аргумент2 [тип_параметра2]
  тип_данных2,
 . . .) ]
 IS | AS
 тело_процедуры;
```


Где *тип_параметра* = IN | OUT | IN OUT

Пример процедуры

```
CREATE OR REPLACE PROCEDURE add_dept
   (p_id      IN dept.deptno%TYPE,
    p_name IN dept.name%TYPE)
IS
  v_cnt BINARY_INTEGER;
BEGIN
  SELECT count(*) INTO v_cnt
      FROM dept WHERE deptno = p_id;
  IF v_cnt = 0 THEN
      INSERT INTO dept(deptno, name) VALUES(p_id, p_name);
  ELSE
      dbms_output.put_line('Такой отдел уже есть в БД');
  END IF;
END;
```

Вызов процедуры

- Неявная **позиционная нотация**

```
BEGIN
   add_dept(50, ‘Отдел АСУ’);
END;
```

- Явная **именованная нотация**

```
BEGIN
   add_dept(p_id => 50, p_name => ‘Отдел АСУ’);
END;
```

Сопровождение процедур

- **Перекомпиляция процедуры**

```
ALTER  PROCEDURE  add_emp  COMPILE;
```

- **Удаление процедуры**

```
DROP  PROCEDURE  add_emp;
```

- **Просмотр текста процедуры**

```
SELECT text FROM user_source
    WHERE name='ADD_EMP'
    ORDER  BY  line;
```

Итоги

- Явные курсоры удобно использовать, когда надо индивидуально обработать каждую запись возвращаемого набора результатов.
- Цикл FOR с курсором наиболее удобная и компактная конструкция для работы с явными курсорами.