

ORACLE 12c

PL/SQL Курсоры

Лекция 10

Курсоры

- ▶ **Курсор** — объект БД, который позволяет приложениям работать с записями построчно
- ▶ Поддерживаются два типа курсоров:
 - ▶ явный — объявляется разработчиком;
 - ▶ неявный — не требует объявления
- ▶ Курсор может возвращать одну строку, несколько строк или ни одной строки
- ▶ Для повторного создания результирующего набора для других значений параметров курсор следует закрыть, а затем повторно открыть



Операторы управления явным курсором

- ▶ DECLARE — выполняет объявление явного курсора.
- ▶ OPEN — открывает курсор, создавая новый результирующий набор на базе указанного запроса.
- ▶ FETCH — выполняет последовательное извлечение строк из результирующего набора от начала до конца.
- ▶ CLOSE — закрывает курсор и освобождает занимаемые им ресурсы.



Атрибуты курсора

- ▶ `%ISOPEN` — возвращает значение TRUE, если курсор открыт
- ▶ `%FOUND` — определяет, найдена ли строка, удовлетворяющая условию
- ▶ `%NOTFOUND` — возвращает TRUE, если строка не найдена
- ▶ `%ROWCOUNT` — возвращает номер текущей строки



Курсоры

- ▶ Курсор Oracle – указатель на область в PGA, в которой хранится:
 - ▶ 1) строки запроса,
 - ▶ 2) число строк,
 - ▶ 3) указатель на разобранный запрос в общем пуле.
- ▶ Открытие курсора – создание контекстной области PGA – создается моментальный снимок (snapshot) данных запроса.



Курсоры

- ▶ PL/SQL позволяет создавать 2 вида курсоров:
 - ▶ 1) Статические курсоры, SQL выражение для которых определяется на этапе компиляции:
 - ▶ Используются для DML команд
 - ▶ Могут быть явно объявлены и именованы
 - ▶ 2) Динамические курсоры, SQL выражение для которых определяется на этапе выполнения:
 - ▶ Могут использоваться для любых SQL выражений, включая DDL и DCL команды
 - ▶ Реализуются с помощью оператора EXECUTE IMMEDIATE или пакета dbms_sql



Неявные курсоры

- ▶ Неявный курсор – выполнение SQL выражения в секции исполнения или в секции исключений блока
- ▶ Операторы INSERT, UPDATE, DELETE, MERGE, SELECT INTO
- ▶ Не требуют объявления
- ▶ Не требуют OPEN, FETCH, CLOSE



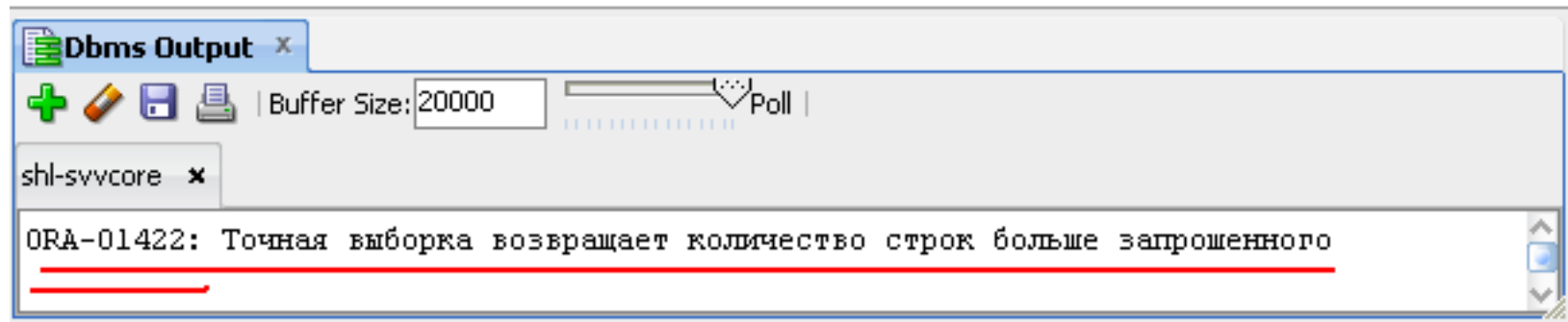
Неявные курсоры

- ▶ Когда неявный курсор не возвращает строк вообще, PL/SQL генерирует исключение `NO_DATA_FOUND` и передает управление в секцию исключений
- ▶ Когда `SELECT` возвращает более одной строки, PL/SQL генерирует исключение `TOO_MANY_ROWS` и также передает управление в секцию исключений
- ▶ `SELECT INTO` предназначен исключительно для того, чтобы возвращать ровно 1 строку – точную выборку



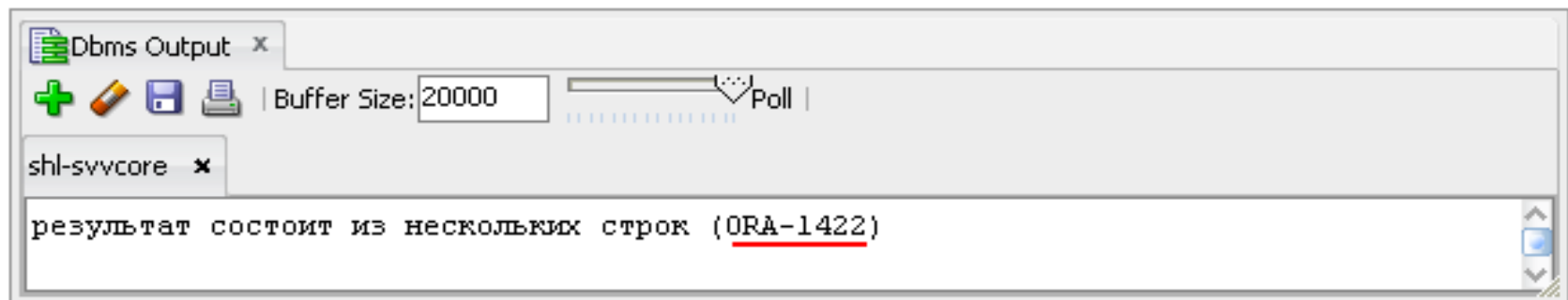
Неявные курсоры

```
-- 11/10.sql  
declare  
    faculty_rec faculty%rowtype;  
begin  
    select * into faculty_rec from faculty;  
    dbms_output.put_line(faculty_rec.faculty || ' ' || faculty_rec.faculty_name);  
exception  
    when others  
    then dbms_output.put_line(sqlerrm);  
  
end;  
/
```



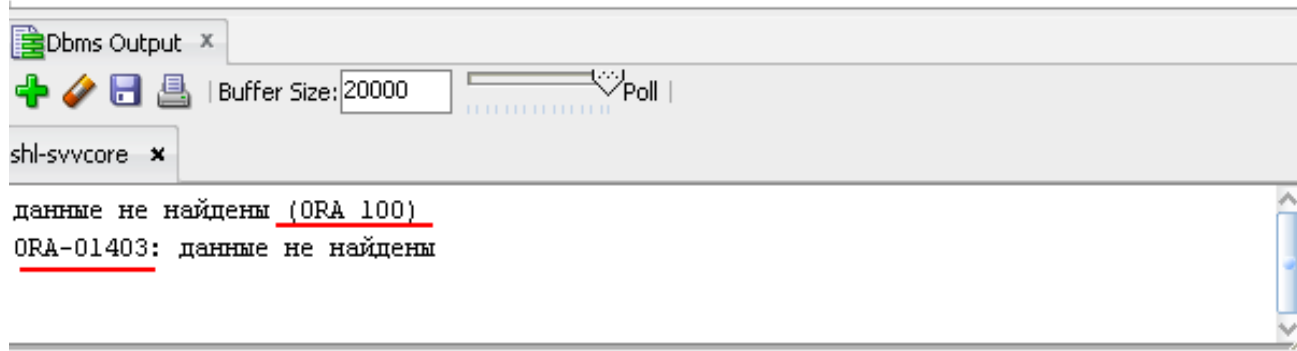
Неявные курсоры

```
declare
    faculty_rec faculty%rowtype;
begin
    select * into faculty_rec from faculty;
    dbms_output.put_line(faculty_rec.faculty || ' ' || faculty_rec.faculty_name);
exception
    when too_many_rows
    then dbms_output.put_line('результат состоит из нескольких строк (ORA' || sqlcode || ')');
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



Неявные курсоры

```
-- 11/17.sql
declare
    faculty_rec faculty%rowtype;
begin
    select * into faculty_rec from faculty where faculty = 'XXX';
    dbms_output.put_line(faculty_rec.faculty || ' ' || faculty_rec.faculty_name);
exception
    when no_data_found
    then
        dbms_output.put_line('данные не найдены (ORA ' || sqlcode || ')');
        dbms_output.put_line(sqlerrm);
    when too_many_rows
    then dbms_output.put_line('результат состоит из нескольких строк (ORA' || sqlcode || ')');
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



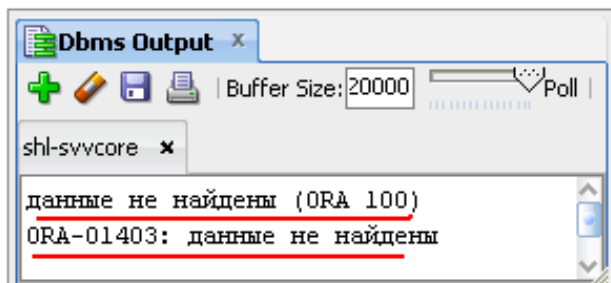
PRAGMA

- ▶ Ключевое слово PRAGMA используется для того, чтобы указать директиву компилятору PL/SQL
- ▶ PRAGMA instruction
- ▶ Не транслируется в исполняемый код
- ▶ Разрешается использовать в секции декларации текущего блока



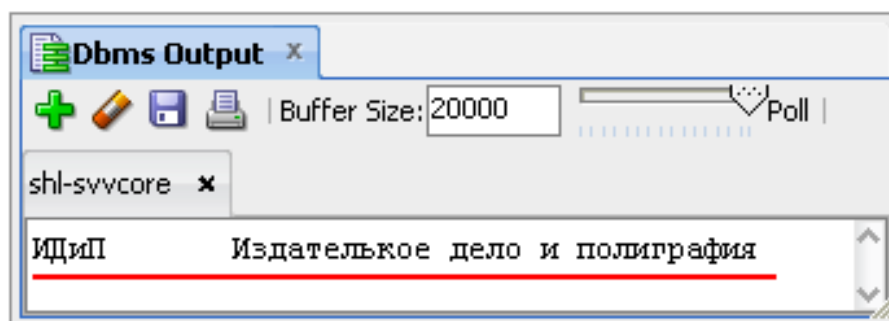
Неявные курсоры – исключения

```
-- 11/17.sql
declare
    faculty_rec faculty%rowtype;
    my_exception exception;
    pragma exception_init(my_exception, 100);
begin
    select * into faculty_rec from faculty where faculty = 'XXX';
    dbms_output.put_line(faculty_rec.faculty || ' ' || faculty_rec.faculty_name);
exception
    when my_exception
    then
        dbms_output.put_line('данные не найдены (ORA ' || sqlcode || ')');
        dbms_output.put_line(sqlerrm);
--when no_data_found
--then
--    dbms_output.put_line('данные не найдены (ORA ' || sqlcode || ')');
--    dbms_output.put_line(sqlerrm);
when too_many_rows
then dbms_output.put_line('результат состоит из нескольких строк (ORA' || sqlcode || ')');
when others
then dbms_output.put_line(sqlerrm);
end;
```



Неявные курсоры

```
-- 11/18.sql
declare
    faculty_rec faculty%rowtype;
begin
    select * into faculty_rec from faculty where faculty = 'ИДИП';
    dbms_output.put_line(faculty_rec.faculty || ' ' || faculty_rec.faculty_name);
exception
    when no_data_found
    then dbms_output.put_line('данные не найдены (ORA ' || sqlcode || ')');
    when too_many_rows
    then dbms_output.put_line('результат состоит из нескольких строк (ORA' || sqlcode || ')');
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



Неявные курсоры – атрибуты курсора

```
-- 11/30.sql
```

```
declare
```

```
    b1    boolean;  
    b2    boolean;  
    b3    boolean;  
    n      pls_integer;  
    faculty_rec faculty%rowtype;
```

```
begin
```

```
select * into faculty_rec from faculty where faculty = 'ИДПИ';
```

```
b1 := sql%found;
```

```
b2 := sql%isopen;
```

```
b3 := sql%notfound;
```

```
n := sql%rowcount;
```

```
dbms_output.put_line(faculty_rec.faculty || ' ' || faculty_rec.faculty_name);
```

```
if b1 then dbms_output.put_line('b1 = TRUE');
```

```
else      dbms_output.put_line('b1 = FALSE');
```

```
end if;
```

```
if b2 then dbms_output.put_line('b2 = TRUE');
```

```
else      dbms_output.put_line('b2 = FALSE');
```

```
end if;
```

```
if b3 then dbms_output.put_line('b3 = TRUE');
```

```
else      dbms_output.put_line('b3 = FALSE');
```

```
end if;
```

```
dbms_output.put_line('n = ' || n);
```

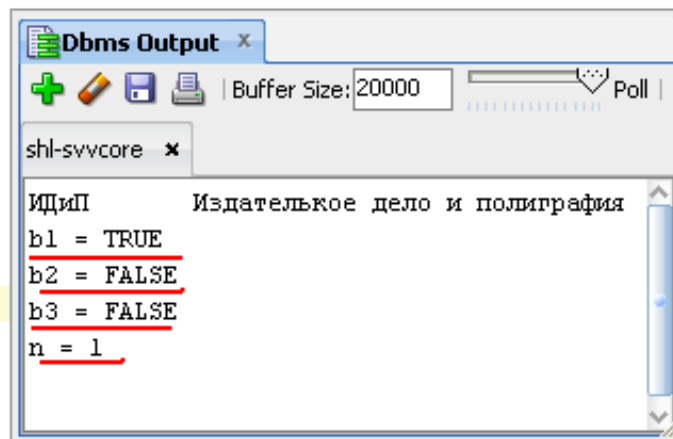
```
exception
```

```
when others
```

```
then dbms_output.put_line(sqlerrm);
```

```
end;
```

```
/
```



Неявные курсоры

Атрибут	Описание
%dSOPEN	Всегда FALSE, так как курсор открывается неявно и закрывается сразу после выполнения
%FOUND	NULL перед выполнением. TRUE, если одна или более строк были вставлены, изменены, удалены или одна строка выбрана. Иначе FALSE
%NOTFOUND	NULL перед выполнением. FALSE, если одна или более строк были вставлены, изменены, удалены или одна строка выбрана. Иначе TRUE
%dROWCOUNT	Количество строк, выбранных в курсоре



Неявные курсоры – UPDATE

```
declare
b1 boolean;
b2 boolean;
b3 boolean;
n pls_integer;
auditorium_cur auditorium%rowtype;
begin
update auditorium set auditorium = '314-1',
                      auditorium_name = '314-1',
                      auditorium_capacity = 90,
                      auditorium_type = 'ЛК'
      where auditorium = '301-1';
b1 := sql%found;
b2 := sql%isopen;
b3 := sql%notfound;
n := sql%rowcount;
dbms_output.put_line(auditorium_cur.auditorium_name || ' ' ||
                      auditorium_cur.auditorium_capacity || ' ' ||
                      auditorium_cur.auditorium_type);
if b1 then dbms_output.put_line('b1 = true');
  else dbms_output.put_line('b1 = false');
end if;
if b2 then dbms_output.put_line ('b2 = true');
  else dbms_output.put_line('b2 = false');
end if;
if b3 then dbms_output.put_line('b3 = true');
  else dbms_output.put_line('b3 = false');
end if;
dbms_output.put_line('n = ' || n);
--commit;
rollback;
exception
  when others
  then dbms_output.put_line (sqlerrm);
end;
```

```
b1 = true
b2 = false
b3 = false
n = 1
```

Неявные курсоры – UPDATE

```
declare
b1 boolean;
b2 boolean;
b3 boolean;
n pls_integer;
auditorium_cur auditorium%rowtype;
begin
update auditorium set auditorium = '314-1',
                      auditorium_name = '314-1',
                      auditorium_capacity = 90,
                      auditorium_type = 'ЖК'
  where auditorium = '301-1';
rollback;
b1 := sql%found;
b2 := sql%isopen;
b3 := sql%notfound;
n := sql%rowcount;
dbms_output.put_line(auditorium_cur.auditorium_name || ' ' ||
                      auditorium_cur.auditorium_capacity || ' ' ||
                      auditorium_cur.auditorium_type);
if b1 then dbms_output.put_line('b1 = true');
  else dbms_output.put_line('b1 = false');
end if;
if b2 then dbms_output.put_line ('b2 = true');
  else dbms_output.put_line('b2 = false');
end if;
if b3 then dbms_output.put_line('b3 = true');
  else dbms_output.put_line('b3 = false');
end if;
dbms_output.put_line('n = ' || n);
---commit;
exception
  when others
  then dbms_output.put_line (sqlerrm);
.
```

```
b1 = false
b2 = false
b3 = true
n = 0
```

Неявные курсоры –INSERT

```
declare
b1 boolean;
b2 boolean;
b3 boolean;
n pls_integer;
auditorium_cur auditorium%rowtype;
begin
insert into auditorium(auditorium, auditorium_name, auditorium_capacity, auditorium_type)
values ('301-1', '301-1', 90, 'JK');
b1 := sql%found;
b2 := sql%isopen;
b3 := sql%notfound;
n := sql%rowcount;
dbms_output.put_line(auditorium_cur.auditorium_name || ' ' ||
                      auditorium_cur.auditorium_capacity || ' ' ||
                      auditorium_cur.auditorium_type);
if b1 then dbms_output.put_line('b1 = true');
           else dbms_output.put_line('b1 = false');
end if;
if b2 then dbms_output.put_line ('b2 = true');
           else dbms_output.put_line('b2 = false');
end if;
if b3 then dbms_output.put_line('b3 = true');
           else dbms_output.put_line('b3 = false');
end if;
dbms_output.put_line('n = ' || n);
--commit;
rollback;
exception
when others
then dbms_output.put_line (sqlerrm);
end;
```

```
b1 = true
b2 = false
b3 = false
n = 1
```

Неявные курсоры –INSERT

```
declare
b1 boolean;
b2 boolean;
b3 boolean;
n pls_integer;
auditorium_cur auditorium%rowtype;
begin
insert into auditorium(auditorium, auditorium_name, auditorium_capacity, auditorium_type)
values ('301-1', '301-1', 90, 'ЖК');
rollback;
b1 := sql%found;
b2 := sql%isopen;
b3 := sql%notfound;
n := sql%rowcount;
dbms_output.put_line(auditorium_cur.auditorium_name || ' ' ||
                    auditorium_cur.auditorium_capacity || ' ' ||
                    auditorium_cur.auditorium_type);
if b1 then dbms_output.put_line('b1 = true');
    else dbms_output.put_line('b1 = false');
end if;
if b2 then dbms_output.put_line ('b2 = true');
    else dbms_output.put_line('b2 = false');
end if;
if b3 then dbms_output.put_line('b3 = true');
    else dbms_output.put_line('b3 = false');
end if;
dbms_output.put_line('n = ' || n);
--commit;
--rollback;
exception
    when others
    then dbms_output.put_line (sqlerrm);
end;
```

```
b1 = false
b2 = false
b3 = true
n = 0
```

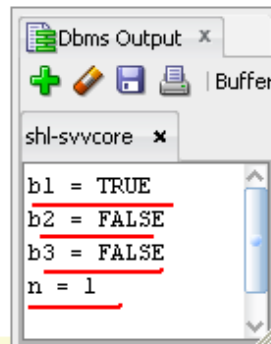
Неявные курсоры –INSERT

```
declare
b1 boolean;
b2 boolean;
b3 boolean;
n pls_integer;
auditorium_cur auditorium%rowtype;
begin
insert into auditorium(auditorium, auditorium_name, auditorium_capacity, audito
values ('301-1', '301-1', 90, 'ЖК');
b1 := sql%found;
b2 := sql%isopen;
b3 := sql%notfound;
n := sql%rowcount;
dbms_output.put_line(auditorium_cur.auditorium_name || ' ' ||
                    auditorium_cur.auditorium_capacity || ' ' ||
                    auditorium_cur.auditorium_type);
if b1 then dbms_output.put_line('b1 = true');
    else dbms_output.put_line('b1 = false');
end if;
if b2 then dbms_output.put_line ('b2 = true');
    else dbms_output.put_line('b2 = false');
end if;
if b3 then dbms_output.put_line('b3 = true');
    else dbms_output.put_line('b3 = false');
end if;
dbms_output.put_line('n = ' || n);
--commit;
rollback;
exception
    when others
    then dbms_output.put_line (sqlerrm);
end;
```

ORA-00001: нарушено ограничение уникальности (SYSTEM.SYS_C009869)

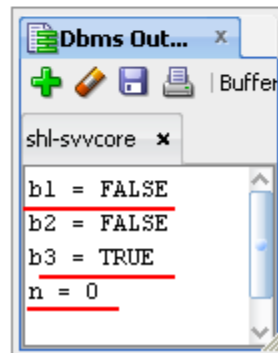
Неявные курсоры – DELETE

```
-- 11/35.sql
declare
    b1    boolean;
    b2    boolean;
    b3    boolean;
    n     pls_integer;
begin
    delete auditorium where auditorium = '301-4';
    b1 := sql%found;
    b2 := sql%isopen;
    b3 := sql%notfound;
    n := sql%rowcount;
    if b1 then dbms_output.put_line('b1 = TRUE');
    else      dbms_output.put_line('b1 = FALSE');
    end if;
    if b2 then dbms_output.put_line('b2 = TRUE');
    else      dbms_output.put_line('b2 = FALSE');
    end if;
    if b3 then dbms_output.put_line('b3 = TRUE');
    else      dbms_output.put_line('b3 = FALSE');
    end if;
    dbms_output.put_line('n = '||n);
    -- commit;
    rollback;
exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



Неявные курсоры – DELETE

```
-- 11/35.sql
declare
    b1    boolean;
    b2    boolean;
    b3    boolean;
    n     pls_integer;
begin
    delete auditorium where auditorium = 'XXXX';
    b1 := sql%found;
    b2 := sql%isopen;
    b3 := sql%notfound;
    n := sql%rowcount;
    if b1 then dbms_output.put_line('b1 = TRUE');
    else      dbms_output.put_line('b1 = FALSE');
    end if;
    if b2 then dbms_output.put_line('b2 = TRUE');
    else      dbms_output.put_line('b2 = FALSE');
    end if;
    if b3 then dbms_output.put_line('b3 = TRUE');
    else      dbms_output.put_line('b3 = FALSE');
    end if;
    dbms_output.put_line('n = '||n);
    -- commit;
    rollback;
exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



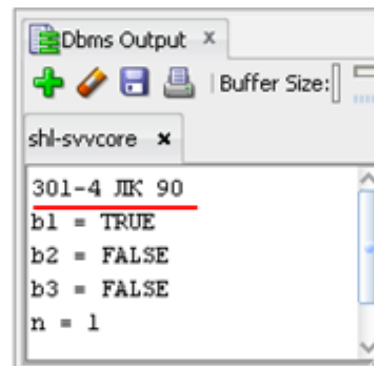
Неявные курсоры – RETURNING

- ▶ RETURNING в выражениях INSERT, UPDATE, DELETE используется для получения данных, измененных соответствующим выражением
- ▶ Позволяет избежать дополнительного SELECT для уточнения результатов



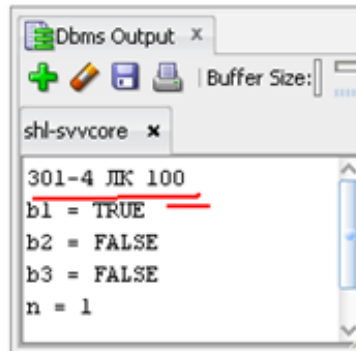
Неявные курсоры – RETURNING

```
declare
    b1    boolean;
    b2    boolean;
    b3    boolean;
    n      pls_integer;
    a_name   svvcore.auditorium.auditorium_name%type;
    a_type   svvcore.auditorium.auditorium_type%type;
    a_capacity svvcore.auditorium.auditorium_capacity%type;
begin
    delete auditorium where auditorium = '301-4'
    returning auditorium_name, auditorium_type, auditorium_capacity
    into a_name, a_type, a_capacity ;
    dbms_output.put_line(a_name||' '||rtrim(a_type)||' '||a_capacity);
    b1 := sql%found;
    b2 := sql%isopen;
    b3 := sql%notfound;
    n := sql%rowcount;
    if b1 then dbms_output.put_line('b1 = TRUE');
    else      dbms_output.put_line('b1 = FALSE');
    end if;
    if b2 then dbms_output.put_line('b2 = TRUE');
    else      dbms_output.put_line('b2 = FALSE');
    end if;
    if b3 then dbms_output.put_line('b3 = TRUE');
    else      dbms_output.put_line('b3 = FALSE');
    end if;
    dbms_output.put_line('n = '||n);
    -- commit;
    rollback;
exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
```



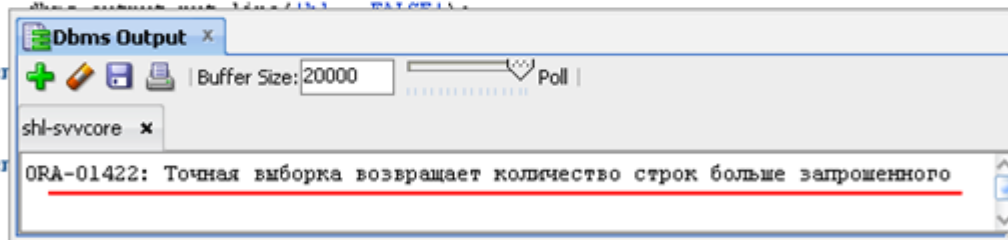
Неявные курсоры – RETURNING

```
declare
    b1    boolean;
    b2    boolean;
    b3    boolean;
    n      pls_integer;
    a_name    svvcore.auditorium.auditorium_name%type;
    a_type    svvcore.auditorium.auditorium_type%type;
    a_capacity svvcore.auditorium.auditorium_capacity%type;
begin
    update auditorium set auditorium_capacity = 100
    where auditorium = '301-4'
    returning auditorium_name, auditorium_type, auditorium_capacity
    into a_name, a_type, a_capacity ;
    dbms_output.put_line(a_name||' '||rtrim(a_type)||' '||a_capacity);
    b1 := sql%found;
    b2 := sql%isopen;
    b3 := sql%notfound;
    n := sql%rowcount;
    if b1 then dbms_output.put_line('b1 = TRUE');
    else      dbms_output.put_line('b1 = FALSE');
    end if;
    if b2 then dbms_output.put_line('b2 = TRUE');
    else      dbms_output.put_line('b2 = FALSE');
    end if;
    if b3 then dbms_output.put_line('b3 = TRUE');
    else      dbms_output.put_line('b3 = FALSE');
    end if;
    dbms_output.put_line('n = '||n);
    -- commit;
    rollback;
exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
```



Неявные курсоры – RETURNING

```
declare
    b1    boolean;
    b2    boolean;
    b3    boolean;
    n      pls_integer;
    a_name svvcore.auditorium.auditorium_name%type;
    a_type svvcore.auditorium.auditorium_type%type;
    a_capacity svvcore.auditorium.auditorium_capacity%type;
begin
    update auditorium set auditorium capacity = 100
    where auditorium_type = 'ЖК'
    returning auditorium name, auditorium type, auditorium capacity
    into a_name, a_type, a_capacity ;
    dbms_output.put_line(a_name||' '||rtrim(a_type)||' '||a_capacity);
    b1 := sql%found;
    b2 := sql%isopen;
    b3 := sql%notfound;
    n := sql%rowcount;
    if b1 then dbms_output.put_line('b1 = TRUE');
    else
    end if;
    if b2 then
    else
    end if;
    if b3 then
    else
    end if;
    dbms_output.put_line('n = '||n);
    -- commit;
    rollback;
    exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
```



Явные курсоры

- ▶ Этапы открытия явного курсора:
 - ▶ разбор [parse]
 - ▶ связывание переменных [bind]
 - ▶ выполнение [execute]
 - ▶ построение плана выполнения запроса [determining execution plan]
 - ▶ связывание внешних переменных [associating host variables] и курсорных параметров
 - ▶ определение набора данных
 - ▶ выставление указателя текущей строки на первую строку в результирующем наборе данных

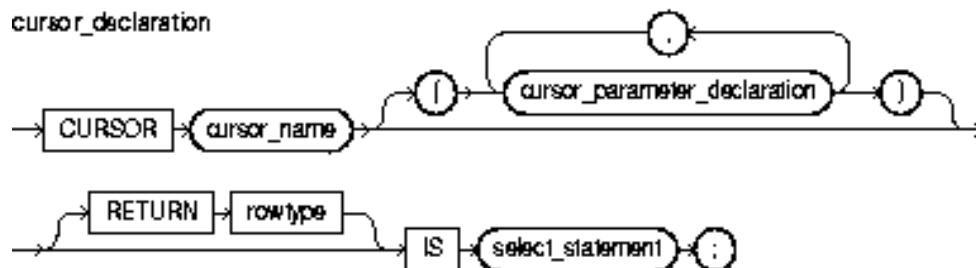


Явные курсоры

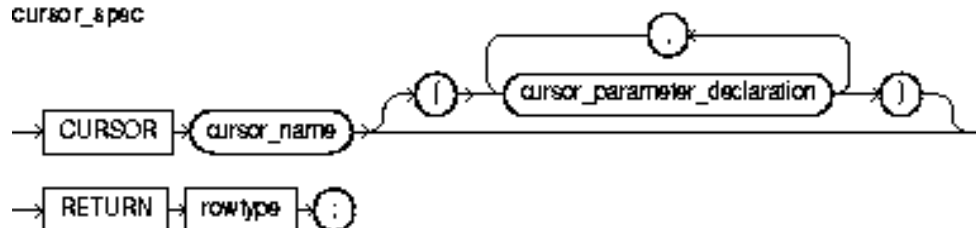
- ▶ Открытие явного курсора:
 - ▶ `OPEN cursor_name [(argument [,argument ...])];`
- ▶ Выборка из явного курсора:
 - ▶ `FETCH cursor_name INTO record or variable_list;`
- ▶ Заккрытие явного курсора:
 - ▶ `CLOSE cursor_name;`



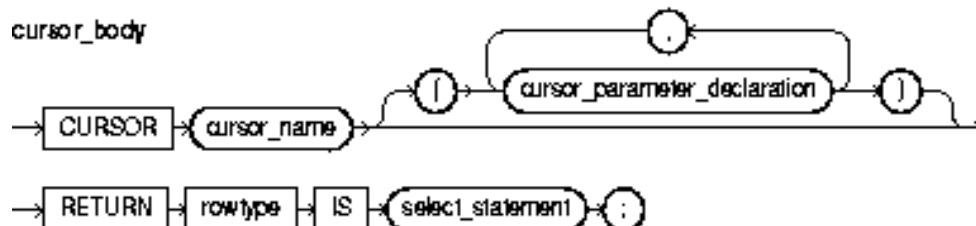
cursor_declaration



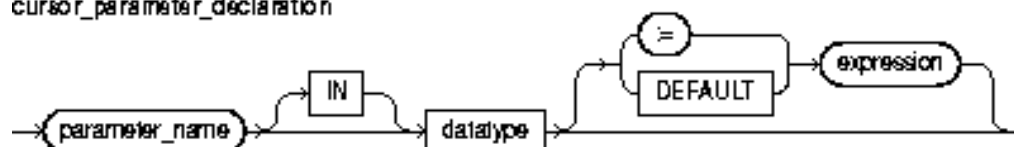
CURSOR_PAGE



cursor_body



cursor_parameter_declaration



Явные курсоры

- ▶ Объявляются явно в секции декларации блока или в спецификации пакета
- ▶ Объявления явного курсора:
 - ▶ Курсор без параметров
 - ▶ `CURSOR company_cur IS SELECT company_id FROM company;`
 - ▶ Курсор, который принимает параметры через список
 - ▶ `CURSOR company__cur (id_in IN NUMBER) IS`
 - ▶ `SELECT name FROM company WHERE company_id = id_in;`
 - ▶ Курсор, который содержит выражение `RETURN` вместо `SELECT`
 - ▶ `CURSOR company_cur (id_in IN NUMBER) RETURN company%ROWTYPE;`

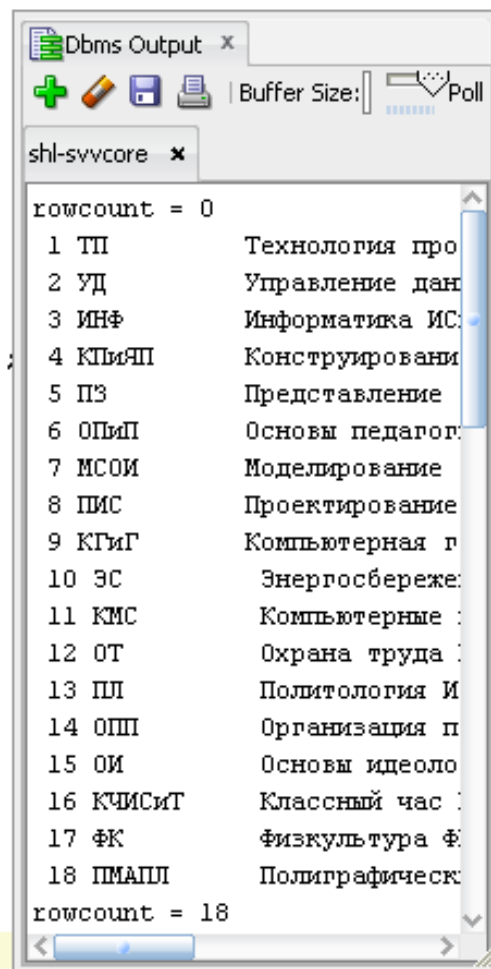


Явные курсоры

```
-- 11/39.sql
declare
  cursor curs_subject is select subject, subject_name, pulpit
    from subject;
  m_subject          svvcore.subject.subject%type;
  m_subject_name     svvcore.subject.subject_name%type;
  m_pulpit           svvcore.subject.pulpit%type;
begin
  open curs_subject;
  dbms_output.put_line('rowcount = ' || curs_subject%rowcount);
  loop
    fetch curs_subject into m_subject, m_subject_name, m_pulpit;
    exit when curs_subject%notfound;
    dbms_output.put_line(' ' || curs_subject%rowcount || ' '
                        || m_subject || ' '
                        || m_subject_name || ' '
                        || m_pulpit);

  end loop;
  dbms_output.put_line('rowcount = ' || curs_subject%rowcount);
  close curs_subject;
exception
  when others
  then dbms_output.put_line(sqlerrm);
end;
```

```
/
```



rowcount = 0	
1	ТП Технология про
2	УД Управление дан
3	ИНФ Информатика ИС
4	КПИАП Конструировани
5	ПЗ Представление
6	ОПИАП Основы педагог
7	МСОИ Моделирование
8	ПИС Проектирование
9	КГиГ Компьютерная п
10	ЭС Энергосбереже
11	КМС Компьютерные :
12	ОТ Охрана труда :
13	ПЛ Политология И
14	ОПП Организация п
15	ОИ Основы идеоло
16	КЧИСиТ Классный час :
17	ФК Физкультура Ф
18	ПМАПЛ Полиграфическ
rowcount = 18	

Явные курсоры

```
-- 11/40.sql
```

```
declare
```

```
    cursor curs_subject is select subject, subject_name, pulpit
                           from subject;
```

```
    rec_subject svvcore.subject%rowtype;
```

```
begin
```

```
    open curs_subject;
```

```
    dbms_output.put_line('rowcount = ' || curs_subject%rowcount);
```

```
    loop
```

```
        fetch curs_subject into rec_subject;
```

```
        exit when curs_subject%notfound;
```

```
        dbms_output.put_line(' ' || curs_subject%rowcount || ' '
                               || rec_subject.subject || ' '
                               || rec_subject.subject_name || ' '
                               || rec_subject.pulpit);
```

```
    end loop;
```

```
    close curs_subject;
```

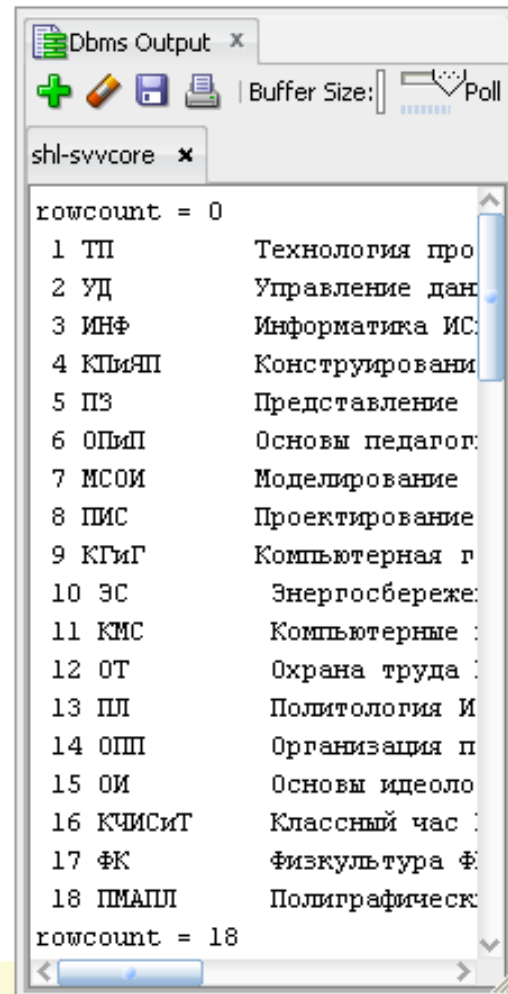
```
    dbms_output.put_line('rowcount = ' || curs_subject%rowcount);
```

```
exception
```

```
    when others
```

```
    then dbms_output.put_line(sqlerrm);
```

```
end;
```

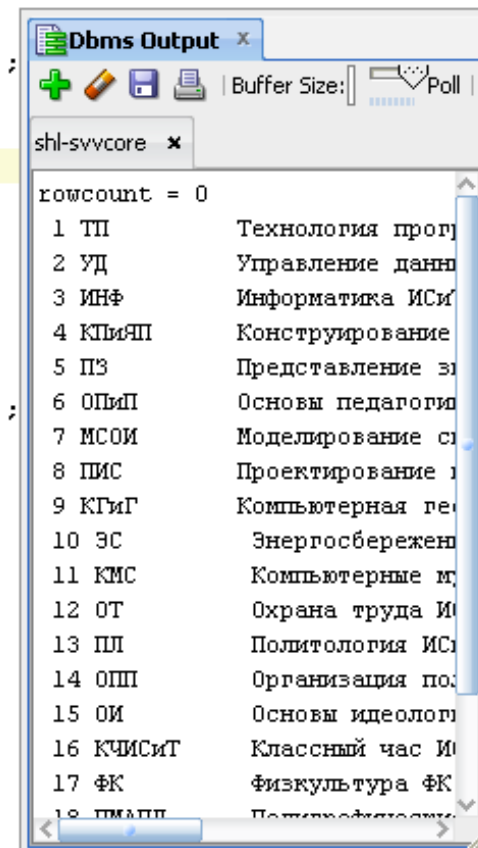


Dbms Output x	
+ Buffer Size: Poll	
shl-svvcore x	
rowcount = 0	
1	ТП Технология про
2	УД Управление дан
3	ИНФ Информатика ИС
4	КПЯП Конструировани
5	ПЗ Представление
6	ОПЯП Основы педагог
7	МСОИ Моделирование
8	ПИС Проектирование
9	КГиГ Компьютерная г
10	ЭС Энергосбереже
11	КМС Компьютерные :
12	ОТ Охрана труда :
13	ПЛ Политология И
14	ОПП Организация п
15	ОИ Основы идеоло
16	КЧИСиТ Классный час :
17	ФК Физкультура Ф
18	ПМАП Полиграфическ
rowcount = 18	

Явные курсоры

```
-- 11/41.sql
declare
    cursor curs_subject is select subject, subject_name, pulpit
                           from subject;
    rec_subject svvcore.subject%rowtype;
begin
    open curs_subject;
    dbms_output.put_line('rowcount = '|| curs_subject%rowcount);
    fetch curs_subject into rec_subject;
    while curs_subject%found
    loop
        dbms_output.put_line(' '||curs_subject%rowcount||' '
                              ||rec_subject.subject||' '
                              || rec_subject.subject_name||' '
                              || rec_subject.pulpit);
        fetch curs_subject into rec_subject;
    end loop;
    dbms_output.put_line('rowcount = '|| curs_subject%rowcount);
    close curs_subject;

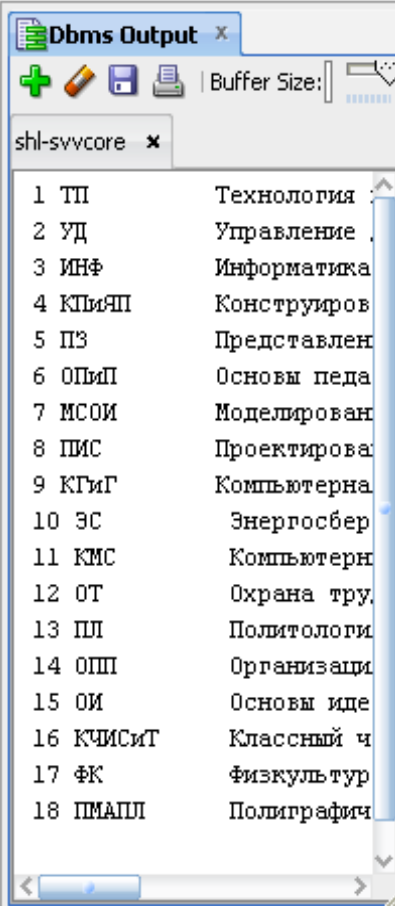
exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



Явные курсоры

```
-- 11/42.sql
declare
    cursor curs_subject is select subject, subject_name, pulpit
                           from subject;
    rec_subject svvcore.subject%rowtype;
begin
    for rec_subject in curs_subject
    loop
        dbms_output.put_line(' ||curs_subject%rowcount|| ' '
                             ||rec_subject.subject|| ' '
                             || rec_subject.subject_name|| ' '
                             || rec_subject.pulpit);
    end loop;

exception
    when others
    then dbms_output.put_line(sqlerrm);
end;
/
```



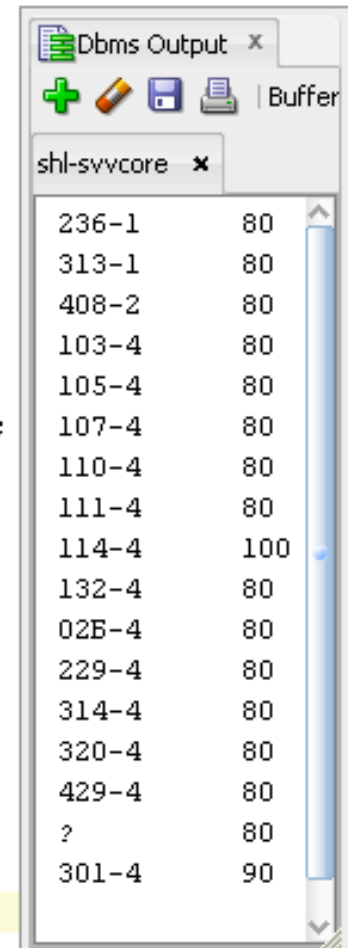
1	ТП	Технология
2	УД	Управление
3	ИНФ	Информатика
4	КПЯП	Конструиров
5	ПЗ	Представлен
6	ОПЯП	Основы педа
7	МСОИ	Моделирован
8	ПИС	Проектирова
9	КГиГ	Компьютерна
10	ЭС	Энергосбер
11	КМС	Компьютерн
12	ОТ	Охрана тру
13	ПИ	Политологи
14	ОПП	Организаци
15	ОИ	Основы иде
16	КЧИСИТ	Классный ч
17	ФК	Физкультур
18	ПМАПЛ	Полиграфия

Явные курсоры с параметрами

```
-- 11/43.sql
declare
  cursor curs_auditorium (capacity svvcore.auditorium.auditorium%type)
  is select auditorium, auditorium_capacity
     from auditorium
     where auditorium_capacity >= capacity;
  -- aum svvcore.auditorium.auditorium%type;
begin
  for aum in curs_auditorium(80)
  loop
    dbms_output.put_line(' ' || aum.auditorium || ' ' || aum.auditorium_capacity);
  end loop;

exception
  when others
  then dbms_output.put_line(sqlerrm);

end;
/
```



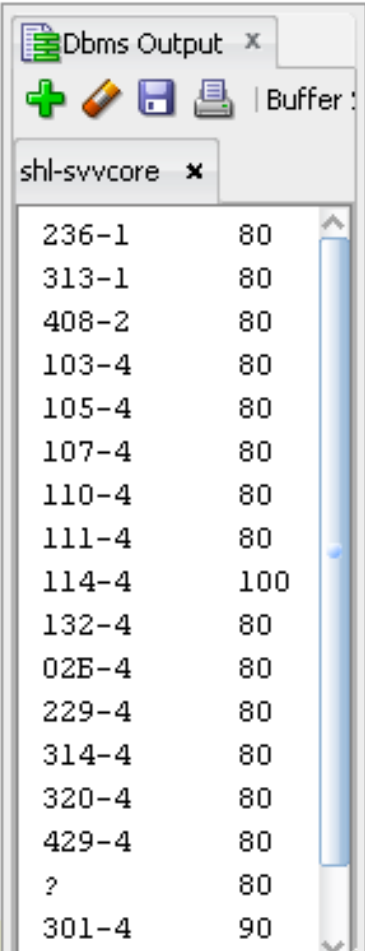
236-1	80
313-1	80
408-2	80
103-4	80
105-4	80
107-4	80
110-4	80
111-4	80
114-4	100
132-4	80
02B-4	80
229-4	80
314-4	80
320-4	80
429-4	80
?	80
301-4	90

Явные курсоры с параметрами

```
-- 11/44.sql
declare
  cursor curs_auditorium (capacity svvcore.auditorium.auditorium%type)
    is select auditorium, auditorium_capacity
       from auditorium
       where auditorium_capacity >= capacity;
  aum svvcore.auditorium.auditorium%type;
  cty svvcore.auditorium.auditorium_capacity%type;
begin
  open curs_auditorium(80);
  fetch curs_auditorium into aum, cty;
  while (curs_auditorium%found)
  loop
    dbms_output.put_line(' ' || aum || ' ' || cty);
    fetch curs_auditorium into aum, cty;
  end loop;

exception
  when others
  then dbms_output.put_line(sqlerrm);

end;
/
```

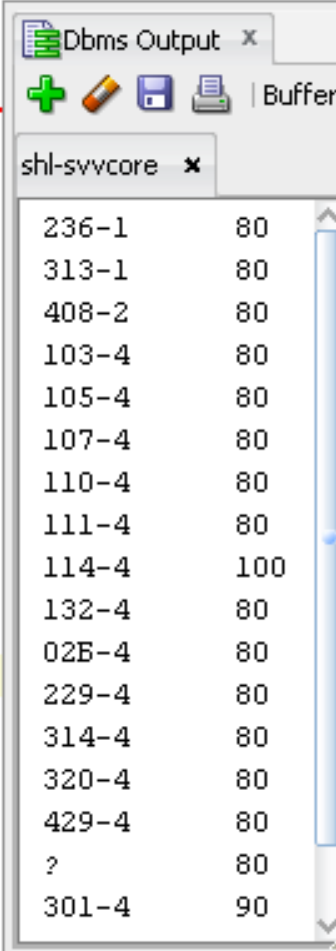


shl-svvcore x	
236-1	80
313-1	80
408-2	80
103-4	80
105-4	80
107-4	80
110-4	80
111-4	80
114-4	100
132-4	80
025-4	80
229-4	80
314-4	80
320-4	80
429-4	80
?	80
301-4	90

Явные курсоры с RETURN

```
-- 11/45.sql
declare
  cursor curs_auditorium (capacity svvcore.auditorium.auditorium%type)
  return auditorium%rowtype
  is select * from auditorium
        where auditorium_capacity >= capacity;
  rec_auditorium svvcore.auditorium%rowtype;
begin
  open curs_auditorium(80);
  fetch curs_auditorium into rec_auditorium;
  while (curs_auditorium%found)
  loop
    dbms_output.put_line(' ||rec_auditorium.auditorium
                        ||' ||rec_auditorium.auditorium_capacity);
    fetch curs_auditorium into rec_auditorium;
  end loop;
  close curs_auditorium;
exception
  when others
  then dbms_output.put_line(sqlerrm);

end;
/
```



236-1	80
313-1	80
408-2	80
103-4	80
105-4	80
107-4	80
110-4	80
111-4	80
114-4	100
132-4	80
02E-4	80
229-4	80
314-4	80
320-4	80
429-4	80
?	80
301-4	90

Явные курсоры

Атрибут	Описание
%dISOPEN	TRUE, если курсор открыт. FALSE, если курсор закрыт.
%FOUND	Ошибка INVALID_CURSOR, если курсор еще не открыт или уже закрыт. NULL перед первой выборкой. TRUE, если запись была успешно выбрана. FALSE, если запись не выбрана.
%NOTFOUND	Ошибка INVALID_CURSOR, если курсор еще не открыт или уже закрыт. NULL перед первой выборкой. FALSE, если запись была успешно выбрана. TRUE, если запись не выбрана.
%dROWCOUNT	Ошибка INVALID_CURSOR, если курсор еще не открыт или уже закрыт. Количество строк, выбранных в курсоре.

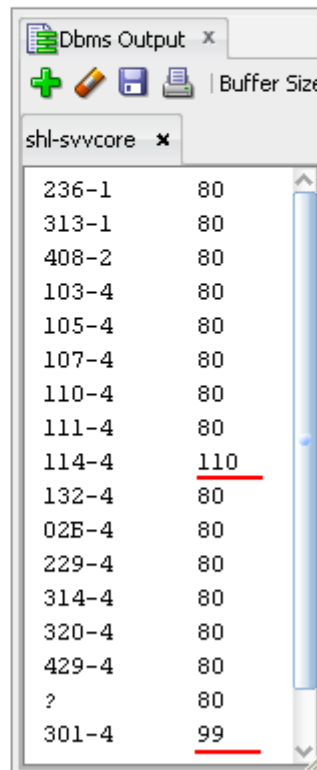


WHERE CURRENT OF

-- 12/50.sql

```
declare
  cursor curs_auditorium (capacity svvcore.auditorium.auditorium%type)
    is select auditorium, auditorium_capacity
       from auditorium
       where auditorium_capacity >= capacity for update;
  aum svvcore.auditorium.auditorium%type;
  cty svvcore.auditorium.auditorium_capacity%type;
begin
  open curs_auditorium(80);
  fetch curs_auditorium into aum, cty;
  while (curs_auditorium%found)
  loop
    if cty >= 90
    then
      cty := cty * 1.1;
      update auditorium
      set auditorium_capacity = cty
      where current of curs_auditorium;
    end if;
    dbms_output.put_line(' '||aum||' '||cty);
    fetch curs_auditorium into aum, cty;
  end loop;
  close curs_auditorium;
  rollback;
exception
  when others
  then dbms_output.put_line(sqlerrm);

end;
```



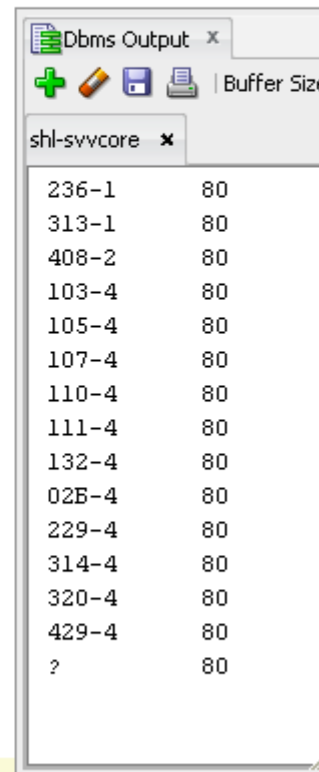
236-1	80
313-1	80
408-2	80
103-4	80
105-4	80
107-4	80
110-4	80
111-4	80
114-4	110
132-4	80
02E-4	80
229-4	80
314-4	80
320-4	80
429-4	80
?	80
301-4	99

WHERE CURRENT OF

```
-- 12/51.sql
declare
  cursor curs_auditorium (capacity svvcore.auditorium.auditorium%type)
    is select auditorium, auditorium_capacity
       from auditorium
       where auditorium_capacity >= capacity for update;
  aum svvcore.auditorium.auditorium%type;
  cty svvcore.auditorium.auditorium_capacity%type;
begin
  open curs_auditorium(80);
  fetch curs_auditorium into aum, cty;
  while (curs_auditorium%found)
  loop
    if cty >= 90
    then
      delete auditorium
      where current of curs_auditorium;
    end if;
    fetch curs_auditorium into aum, cty;
  end loop;
  close curs_auditorium;

  for pp in curs_auditorium(80)
  loop
    dbms_output.put_line(' ||pp.auditorium||
                        ' ||pp.auditorium_capacity);

  end loop;
  rollback;
exception
  when others
  then dbms_output.put_line(sqlerrm);
end;
```



236-1	80
313-1	80
408-2	80
103-4	80
105-4	80
107-4	80
110-4	80
111-4	80
132-4	80
02E-4	80
229-4	80
314-4	80
320-4	80
429-4	80
?	80

Использование ROWID

```
-- 12/52.sql
```

```
select rtrim(tlesson), rowid,  
       substr(rowid, 1,7) "сегмент",  
       substr(rowid,8,3) "№ файла",  
       substr(rowid,11,6) "№ блока",  
       substr(rowid,16,3) "№ строки"  
from tlesson  
/
```

Query Result x

SQL | All Rows Fetched: 9 in 0,015 seconds

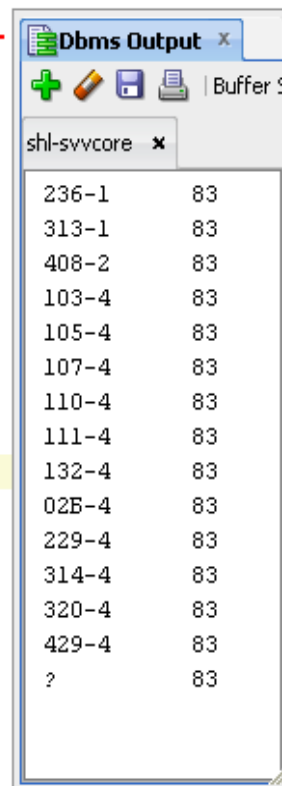
RTRIM(TLESSON)	ROWID	сегмент	№ файла	№ блока	№ строки
ДП	AAAPkWAANAAAAADqAAF	AAAPkWA	АНА	AAADqA	AAF
ИП	AAAPkWAANAAAAADqAAH	AAAPkWA	АНА	AAADqA	AAH
КП	AAAPkWAANAAAAADqAAE	AAAPkWA	АНА	AAADqA	AAE
КЧ	AAAPkWAANAAAAADqAAI	AAAPkWA	АНА	AAADqA	AAI
ЛК	AAAPkWAANAAAAADqAAA	AAAPkWA	АНА	AAADqA	AAA
ЛР	AAAPkWAANAAAAADqAAB	AAAPkWA	АНА	AAADqA	AAB
ПЗ	AAAPkWAANAAAAADqAAC	AAAPkWA	АНА	AAADqA	AAC
ПП	AAAPkWAANAAAAADqAAG	AAAPkWA	АНА	AAADqA	AAG
ФК	AAAPkWAANAAAAADqAAD	AAAPkWA	АНА	AAADqA	AAD

Использование ROWID

```
-- 12/53.sql
declare
  cursor curs_auditorium (capacity svvcore.auditorium.auditorium%type)
    is select auditorium, auditorium_capacity, rowid
       from auditorium
       where auditorium_capacity >= capacity for update;
  aum svvcore.auditorium.auditorium%type;
  cty svvcore.auditorium.auditorium_capacity%type;
begin

  for xxx in curs_auditorium(80)
  loop
    case
      when xxx.auditorium_capacity >= 90
      then delete auditorium
           where rowid = xxx.rowid;
      when xxx.auditorium_capacity >= 80
      then update auditorium
           set auditorium_capacity = auditorium_capacity+3
           where rowid = xxx.rowid;
    end case;
  end loop;
  for yyy in curs_auditorium(80)
  loop
    dbms_output.put_line(' |||yyy.auditorium||
                          ' |||yyy.auditorium_capacity);

  end loop;
  rollback;
exception
  when others
  then dbms_output.put_line(sqlerrm);
end;
/
```



236-1	83
313-1	83
408-2	83
103-4	83
105-4	83
107-4	83
110-4	83
111-4	83
132-4	83
02E-4	83
229-4	83
314-4	83
320-4	83
429-4	83
?	83

Использование ROWNUM

► ROWNUM - логический номер записи в запросе

-- 12/54.sql

declare

cursor curs_auditorium (capacity auditorium.auditorium***type**, n **number**)

is select auditorium, auditorium_capacity

from (

select auditorium, auditorium_capacity

from auditorium

where auditorium capacity >= capacity

order by auditorium_capacity desc

)

where rownum <= n;

begin

for yyy **in** curs_auditorium(80, 10)

loop

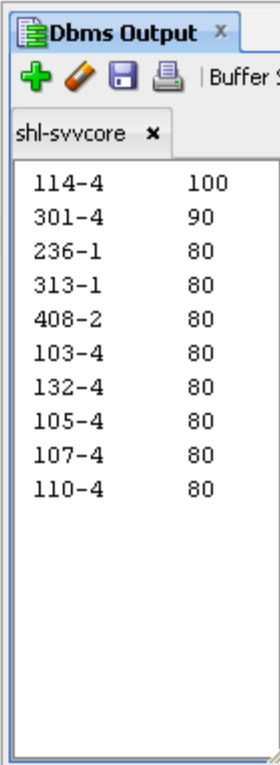
dbms_output.put_line(' ||yyy.auditorium||
' ||yyy.auditorium_capacity);

end loop;

exception

when others then dbms_output.put_line(sqlerrm);

end;



114-4	100
301-4	90
236-1	80
313-1	80
408-2	80
103-4	80
132-4	80
105-4	80
107-4	80
110-4	80

SELECT FOR UPDATE

- ▶ FOR UPDATE используется для блокировки строк в результирующем наборе
- ▶ Использование FOR UPDATE не требует действительного изменения данных
- ▶ Блокировки освобождаются по завершению транзакции
- ▶ Синтаксис:
 - ▶ `SELECT ... FROM ... FOR UPDATE [OF column_reference] [NOWAIT];`
- ▶ NOWAIT – не ожидать освобождения других блокировок



SELECT FOR UPDATE

```
❏ declare
  cursor STUDENT_GRADE_CUR is
  select NAMESTUDENT, GRADEDATE, GRADE
  from STUDENT inner join STUDENT_GRADES
  on STUDENT.STUDENT = STUDENT_GRADES.IDSTUDENT
  for update of STUDENT_GRADES.GRADEDATE, STUDENT_GRADES.GRADE;
begin
  null;
end;
```



Курсорные переменные

- ▶ Курсорные переменные - это структуры данных, которые указывают на курсорный объект
- ▶ Используются для:
 - ▶ Передачи курсора в качестве параметра,
 - ▶ Чтобы отложить связь курсора с SELECT-запросом до выполнения команды OPEN



Курсорные переменные

-- 11/46.sql

declare

type tlesson type is ref cursor return tlesson%rowtype;

xcurs tlesson_type;

rec_tlesson tlesson%rowtype;

begin

open xcurs **for** **select** * **from** svvcore.tlesson;

fetch xcurs **into** rec_tlesson;

while (xcurs%found)

loop

dbms_output.put_line(' '||rtrim(rec_tlesson.tlesson)
||' '||rec_tlesson.tlesson_name);

fetch xcurs **into** rec_tlesson;

end loop;

close xcurs;

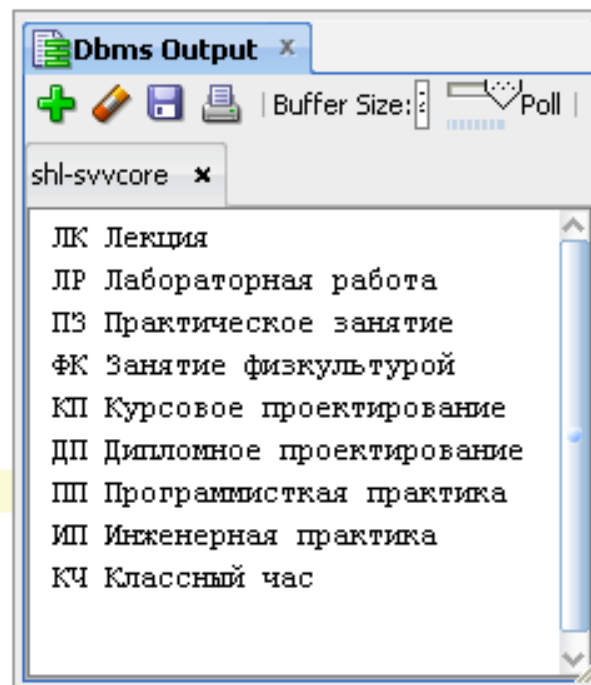
exception

when others

then dbms_output.put_line(sqlerrm);

end;

/



Курсорные переменные

- ▶ Курсорная переменная, объявленная с помощью REF CURSOR без указания RETURN может быть связана с любым запросом
- ▶ Курсорная переменная, объявленная с помощью REF CURSOR с указанием RETURN может быть связана только с запросом, который возвращает результат точно соответствующий числу и типам данных в записи после фразы RETURN во время выполнения



Курсорные переменные

```
SVUCORE@sh1> variable x refcursor;
SVUCORE@sh1> edit
Записано file afiedt.buf

 1  declare
 2      type tlesson_type is ref cursor return tlesson%rowtype;
 3      xcurs tlesson_type;
 4      rec_tlesson tlesson%rowtype;
 5      begin
 6      open xcurs for select * from svucore.tlesson;
 7      :x := xcurs;
 8  exception
 9      when others
10      then dbms_output.put_line(sqlerrm);
11* end;
SVUCORE@sh1> /
```

Процедура PL/SQL успешно завершена.

```
SVUCORE@sh1> print x;
```

TLESSON	TLESSON_NAME
ЛК	Лекция
ЛР	Лабораторная работа
ПЗ	Практическое занятие
ФК	Занятие физкультурой
КП	Курсовое проектирование
ДП	Дипломное проектирование
ПП	Программисткая практика
ИП	Инженерная практика
КЧ	Классный час

9 строк выбрано.

Курсорные переменные

```
-- 11/47.sql
```

```
variable x refcursor;
```

```
declare
```

```
    type tlesson_type is ref cursor return tlesson%rowtype;
```

```
    xcurs tlesson_type;
```

```
    rec_tlesson tlesson%rowtype;
```

```
begin
```

```
    open xcurs for select * from   svwcore.tlesson;
```

```
    :x := xcurs;
```

```
exception
```

```
    when others
```

```
    then dbms_output.put_line(sqlerrm);
```

```
end;
```

```
/
```

```
print x;
```

```
anonymous block completed
```

```
X
```

```
-----  
TLESSON    TLESSON_NAME  
-----
```

ЛК	Лекция
ЛР	Лабораторная работа
ПЗ	Практическое занятие
ФК	Занятие физкультурой
КП	Курсовое проектирование
ЦП	Дипломное проектирование
ПП	Программисткая практика
ИП	Инженерная практика

Курсорные переменные

declare

```
xcurs sys_refcursor;  
rec_tlesson tlesson%rowtype;
```

begin

```
open xcurs for select * from svvcore.tlesson;
```

```
fetch xcurs into rec_tlesson;
```

```
while (xcurs%found)
```

loop

```
dbms_output.put_line(' '||rtrim(rec_tlesson.tlesson)  
                    ||' '||rec_tlesson.tlesson_name);
```

```
fetch xcurs into rec_tlesson;
```

end loop;

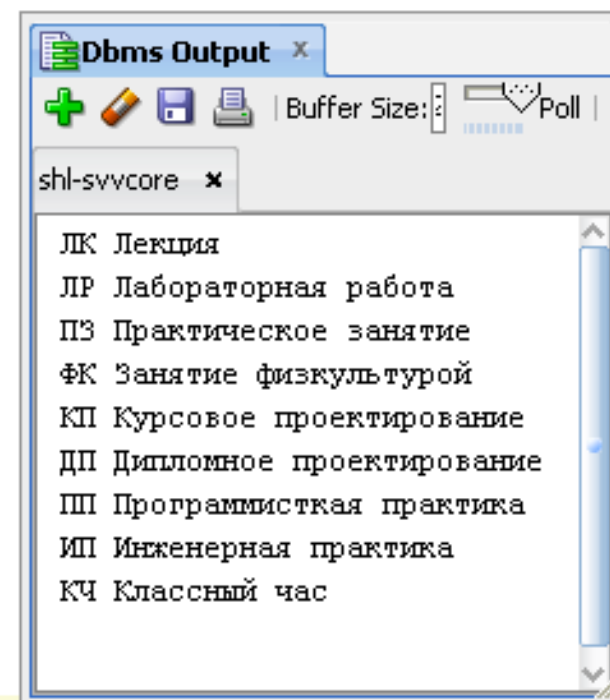
```
close xcurs;
```

exception

when others

```
then dbms_output.put_line(sqlerrm);
```

```
end;
```



Курсорные подзапросы

```
declare
  cursor curs aut
  is select auditorium_type,
         cursor {
           select auditorium
             from auditorium aum
            where aut.auditorium_type = aum.auditorium_type
         }
         from auditorium_type aut;
  curs_aum sys_refcursor;
  aut auditorium_type.auditorium_type%type;
  txt varchar2(1000);
  aum auditorium.auditorium%type;
begin
  open curs_aut;
  fetch curs_aut into aut, curs_aum;
  while (curs_aut%found)
  loop
    txt := rtrim(aut)||':';
    loop
      fetch curs_aum into aum;
      exit when curs_aum%notfound;
      txt := txt||','||rtrim(aum);
    end loop;
    dbms_output.put_line(txt);
    fetch curs_aut into aut, curs_aum;
  end loop;
  close curs_aut;
exception
  when others
  then dbms_output.put_line(sqlerrm);
end;
```

```
ЛК: ,236-1,313-1,324-1,408-2,103-4,105-4,107-4,110-4,111-4,132-4,02Б-4,229-4,314-4,
ЛБ-К: ,206-1,301-1,413-1,423-1,304-4
ЛК-К: ,114-4
ЛБ-Х:
ЛБ-СК:
```

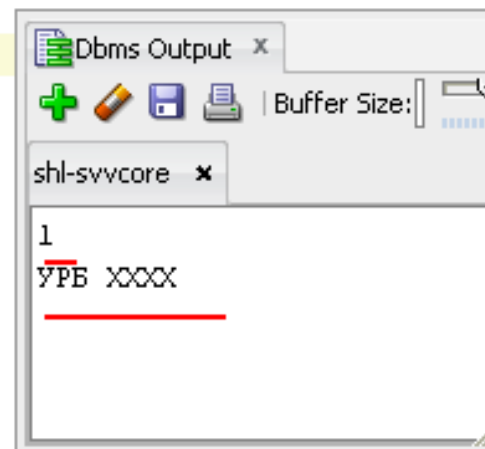
Динамические курсоры

- ▶ EXECUTE IMMEDIATE - однострочные запросы и DDL команды
- ▶ OPEN FOR, FETCH и CLOSE - динамические многострочные запросы
- ▶ Синтаксис EXECUTE IMMEDIATE:
 - ▶ EXECUTE IMMEDIATE sql_statement
 - ▶ [INTO {variable [,variable ...] | record}] [USING [IN | OUT | IN OUT] bind_argument
 - ▶ [, [IN | OUT | IN OUT] bind_argument . ..]] [{RETURNING | RETURN} INTO bind_argument [,bind_argument]...];



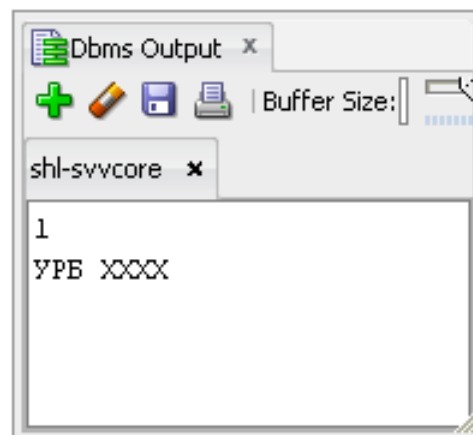
Динамические курсоры

```
-- 14/08.sql
declare
    stmt varchar2(150) := 'update teacher set teacher name = :1 where teacher = :2';
    t teacher.teacher%type;
    tn teacher.teacher_name%type;
    rec teacher%rowtype;
begin
    t := 'YPE'; tn := 'XXXX';
    execute immediate stmt using tn, t;
    dbms_output.put_line(sql%rowcount);
    select * into rec from teacher where teacher = 'YPE';
    dbms_output.put_line(rtrim(rec.teacher) || ' ' || rec.teacher_name);
exception
    when others then
        dbms_output.put_line(sqlerrm);
end;
```



Динамические курсоры

```
-- 14/09.sql
declare
    stmt varchar2(150) := 'begin update teacher set teacher_name = :1 where teacher = :2;commit;end;';
    t teacher.teacher%type;
    tn teacher.teacher_name%type;
    rec teacher%rowtype;
begin
    t := 'YPB'; tn := 'XXXX';
    execute immediate stmt using tn, t;
    dbms_output.put_line(sql%rowcount);
    select * into rec from teacher where teacher = 'YPB';
    dbms_output.put_line(rtrim(rec.teacher) || ' ' || rec.teacher_name);
exception
    when others then
        dbms_output.put_line(sqlerrm);
end;
```



Динамические курсоры

- ▶ Для улучшения производительности выполнения SQL выражений можно использовать динамические курсоры со связанными переменными
- ▶ Это позволяет серверу Oracle повторно использовать разобранные SQL выражения из разделяемого пула
- ▶ EXECUTE IMMEDIATE 'INSERT INTO
- ▶ dept (deptno, dname, loc) VALUES (:deptno, :dname, :loc)'
USING deptno_in, dname_in, loc_in;



Динамические курсоры - OPEN FOR

```
DECLARE
  TYPE cv_typ IS REF CURSOR;
  cv cv_typ;
  laccount_no NUMBER;
  lbalance NUMBER;
BEGIN
  OPEN cv FOR
    'SELECT account_no, balance FROM accounts
     WHERE balance < 500';
LOOP
  FETCH cv INTO laccount_no, lbalance;
  EXIT WHEN cv%NOTFOUND;
  -- Process the row.
END LOOP;
  CLOSE cv;
END;
```

Параметры Oracle, связанные с курсорами

- ▶ `cursor_space_for_time = {TRUE|FALSE}` – большой объем памяти для курсоров и никогда не освобождается. Применяется для увеличения скорости работы курсоров при наличии памяти для разделяемого пула.
- ▶ `cursor_sharing = {EXACT|SIMILAR|FORCE}`
- ▶ `open_cursors` - максимальное количество открытых курсоров.
- ▶ `session_cached_cursors` – максимальное количество кэшируемых курсоров для сессии.



Параметры Oracle, связанные с курсорами

```
SYSTEM@sh1> show parameter cursor;
```

NAME_COL_PLUS_SHOW_PARAM	TYPE	VALUE_COL
<u>cursor_sharing</u>	string	EXACT
<u>cursor_space_for_time</u>	boolean	FALSE
<u>open_cursors</u>	integer	300
<u>session_cached_cursors</u>	integer	20

```
SYSTEM@sh1>
```

```
SYSTEM@sh1> connect svucore/svucore@sh1;
```

Соединено.

sh1 - SVUCORE - 23.11.10

```
SVUCORE@sh1> alter session set session_cached_cursors=50;
```

Сеанс изменен.

```
SVUCORE@sh1> alter session set cursor_sharing=similar  
2 /
```

Сеанс изменен.

```
SVUCORE@sh1> select * from sys.V_$session_cursor_cache;
```

MAXIMUM	COUNT	OPENED_ONCE	OPEN	OPENS	HITS	HIT_RATIO
<u>50</u>	1	1	0	3	3	1

Вопросы?

