**Оглавление**

[PL/SQL Cursors](#_gjdgxs)

[Implicit Cursor](#_30j0zll)

[Explicit cursor](#_1fob9te)

[Open Cursor](#_3znysh7)

[Close Cursor](#_2et92p0)

[Fetch from cursor](#_tyjcwt)

[Атрибуты курсора](#_3dy6vkm)

[Select into](#_1t3h5sf)

[Cursor for loop statement](#_4d34og8)

[Cursor variable (REF CURSOR)](#_2s8eyo1)

[Операции с cursor variable](#_17dp8vu)

[Объявление курсорной переменной](#_3rdcrjn)

[Открытие и закрытие](#_26in1rg)

[Извлечение данных](#_lnxbz9)

[Связывание значения с курсорной переменной](#_35nkun2)

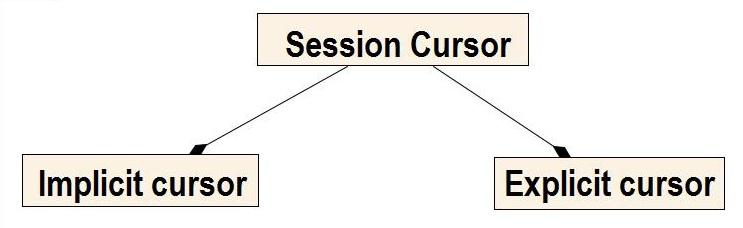
[Cursor expression](#_1ksv4uv)

[Параметры БД](#_44sinio)

[Словари данных](#_2jxsxqh)

# PL/SQL Cursors

**Курсор** – указатель на приватную SQL область, в которой содержится информация о выполнении конкретного select-оператора (или любого другого dml оператора) .



## Implicit Cursor

* PL/SQL открывает implicit cursor каждый раз, когда выполняет select или dml-оператор. Сразу после выполнения закрывает его.
* Мы не контролируем такие курсоры
* К некоторым атрибутам таких курсоров мы имеем доступ

|  |  |
| --- | --- |
| **Атрибут** | **Описание** |
| SQL%ISOPEN | Всегда возвращает false |
| SQL%FOUND | Возвращает null, если никакого запроса не было выполнено; true, если запрос вернул какие-то записи (или повлиял на какие-то записи); иначе false |
| SQL%NOTFOUND | Возвращает null, если никакого запроса не было выполнено; false, если запрос вернул какие-то записи (или повлиял на какие-то записи); иначе true |
| SQL%ROWCOUNT | Количество записей, которые вернул запрос (на которые повлиял запрос) |

**Пример**:

**CREATE OR REPLACE PROCEDURE p(dept\_no NUMBER) AUTHID DEFINER AS**

**BEGIN**

**DELETE FROM dept\_temp**

**WHERE department\_id = dept\_no;**

**IF SQL%FOUND**

**THEN**

**DBMS\_OUTPUT.PUT\_LINE('Delete succeeded for department number ' || dept\_no);**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE('No department number ' || dept\_no);**

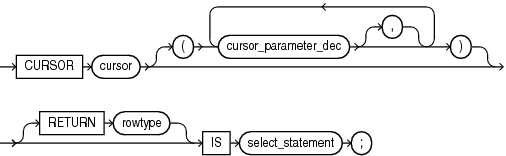
**END IF;**

**END;**

## Explicit cursor

* Это именованный курсор (named cursor).
* Объявление курсора

CURSOR *cursor\_name* [ *parameter\_list* ] [ RETURN *return\_type* ] IS *select\_statement*;



**Restrictions**:

* Select-оператор не может содержать конструкцию with

**Пример**:

**DECLARE**

**CURSOR c1 IS**

**SELECT employee\_id,**

**job\_id,**

**salary**

**FROM employees**

**WHERE salary > 2000;**

**BEGIN**

**some\_actions;**

**END;**

Операции с курсором:

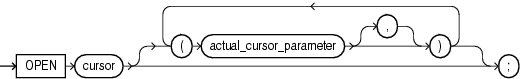
* Open cursor
* Close cursor
* Fetch from cursor

### Open Cursor

1. Выделяет ресурсы базы данных для выполнения запроса

2. Выполняет запрос

3. Устанавливает позицию курсора перед первой записью результирующего набора

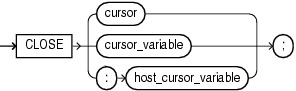


**Возможные** **Exceptions**

* INVALID\_CURSOR
* CURSOR\_ALREADY\_OPEN

### Close Cursor

Освобождает все ресурсы, связанные с данным курсором



**Возможные** **Exceptions**

* INVALID\_CURSOR
* CURSOR\_ALREADY\_OPEN

### Fetch from cursor

FETCH *cursor\_name* INTO *into\_clause*

1. Извлекает текущую строку из курсора
2. Сохраняет значение колонок в переданных переменных
3. Переводит позицию курсора на следующую строку



* В операторе часто используются переменные, объявленные с помощью динамического типа %TYPE и %ROWTYPE
* Если в курсоре одно из полей – это некоторое выражение, для этого поля необходим псевдоним в двух случаях: если данные фетчатся в переменную типа %ROWTYPE, или если мы обращаемся к полю по имени

**Пример** (обратите внимание на использование внешних переменных в курсоре):

**DECLARE**

**sal employees.salary%TYPE;**

**sal\_multiple employees.salary%TYPE;**

**factor INTEGER := 2;**

**CURSOR c1 IS**

**SELECT salary,**

**salary \* factor**

**FROM employees**

**WHERE job\_id LIKE 'AD\_%';**

**BEGIN**

**OPEN c1;**

**LOOP**

**FETCH c1**

**INTO sal, sal\_multiple;**

**EXIT WHEN c1%NOTFOUND;**

**DBMS\_OUTPUT.PUT\_LINE('factor = ' || factor);**

**DBMS\_OUTPUT.PUT\_LINE('sal = ' || sal);**

**DBMS\_OUTPUT.PUT\_LINE('sal\_multiple = ' || sal\_multiple);**

**factor := factor + 1;**

**END LOOP;**

**CLOSE c1;**

**END;**

**Курсор может принимать параметры**:

**DECLARE**

**CURSOR c(job VARCHAR2,**

**max\_sal NUMBER,**

**hired DATE DEFAULT '31-DEC-99') IS**

**SELECT last\_name,**

**first\_name,**

**(salary - max\_sal) overpayment**

**FROM employees**

**WHERE job\_id = job**

**AND salary > max\_sal**

**AND hire\_date > hired**

**ORDER BY salary;**

**BEGIN**

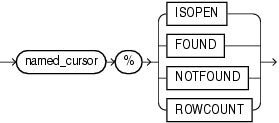
**OPEN c('SA\_REP', 10000, '31-DEC-04');**

**do\_something;**

**CLOSE c;**

**END;**

**Атрибуты курсора**:



|  |  |
| --- | --- |
| **Атрибут** | **Описание** |
| %ISOPEN | Возвращает true, если курсор открыт; иначе false. Xаще всего используется для проверки перед открытием или закрытием курсора |
| %FOUND | Возвращает null, если курсор открыт, но не было ни одного fetch; true, если последний fetch вернул записи; и false, если последний fetch не вернул ни одной записи |
| %NOTFOUND | Возвращает null, если курсор открыт, но не было ни одного fetch; false, если последний fetch вернул записи; и true, если последний fetch не вернул ни одной записи |
| %ROWCOUNT | Возвращает суммарное количество извлеченных записей |

Для закрытого курсора обращение к любому атрибуту, кроме %ISOPEN, генерирует исключение **INVALID\_CURSOR**

**Пример**:

**DECLARE**

**CURSOR c1 IS**

**SELECT last\_name,**

**salary**

**FROM employees**

**WHERE ROWNUM < 11**

**ORDER BY last\_name;**

**my\_ename employees.last\_name%TYPE;**

**my\_salary employees.salary%TYPE;**

**BEGIN**

**OPEN c1;**

**LOOP**

**FETCH c1**

**INTO my\_ename,**

**my\_salary;**

**IF c1%NOTFOUND**

**THEN**

**EXIT;**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE('Name = ' || my\_ename || ', salary = ' || my\_salary);**

**END IF;**

**END LOOP;**

**END;**

# Select into

**Получение одной записи:**

**SELECT select\_item [, select\_item ]...**

**INTO variable\_name [, variable\_name ]...**

**FROM table\_name;**

**Получение нескольких записей:**

**SELECT select\_item [, select\_item ]...**

**BULK COLLECT INTO variable\_name [, variable\_name ]...**

**FROM table\_name;**

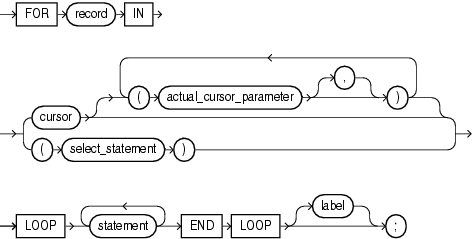
**Возможные Exceptions:**

NO\_DATA\_FOUND

TOO\_MANY\_ROWS

# Cursor for loop statement

* Используется implicit cursor (в этом случае называется implicit cursor FOR LOOP statement)
* На такой курсор нельзя ссылаться с помощью SQL
* В конструкции можно использовать explicit cursor (в этом случае называется explicit cursor FOR LOOP statement)
* В цикле неявно объявляется переменная типа cursor%ROWTYPE
* Переменная является локальной для цикла, доступна только в теле цикла и живет пока цикл выполняется
* Открывается курсор автоматически
* Закрывается также автоматически
* Курсор закрывается тогда, когда все записи выбраны или если выражение внутри цикла передает управление наружу цикла, либо же если возникает исключение
* Допустимы также курсоры с параметрами
* Каждый fetch выбирает неявно по 100 записей для улучшения производительности (начиная с 10 версии, до этого возвращалось по одной записи)



**Пример**:

**DECLARE**

**CURSOR c1 IS**

**SELECT last\_name,**

**job\_id**

**FROM employees**

**WHERE job\_id LIKE '%CLERK%'**

**AND manager\_id > 120**

**ORDER BY last\_name;**

**BEGIN**

**FOR item IN (SELECT last\_name,**

**job\_id**

**FROM employees**

**WHERE job\_id LIKE '%CLERK%'**

**AND manager\_id > 120**

**ORDER BY last\_name)**

**LOOP**

**DBMS\_OUTPUT.PUT\_LINE('Name = ' || item.last\_name || ', Job = ' || item.job\_id);**

**END LOOP;**

**FOR item IN c1**

**LOOP**

**DBMS\_OUTPUT.PUT\_LINE('Name = ' || item.last\_name || ', Job = ' || item.job\_id);**

**END LOOP;**

**END;**

# Cursor variable (REF CURSOR)

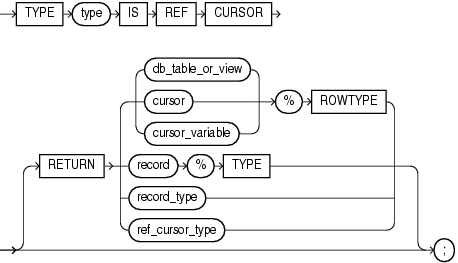
Cursor variable – это указатель, т.е. содержит адрес объекта(курсора), а не сам объект.

* Может использоваться для выполнения разных select-запросов
* Может участвовать в выражениях
* Может быть входным параметром
* Может быть параметром, передающимся от БД клиенту
* Не может содержать параметров
* **Сильный курсор –** если для него определен тип возвращаемого значения. К такому курсору можно привязывать только те запросы, которые возвращают набор данных определенной структуры.
* **Слабый курсор (SYS\_REFCURSOR) -** курсор, для которого тип возвращаемого значения не определен.К нему можно привязывать любые запросы.

## Операции с cursor variable

### Объявление курсорной переменной

**TYPE *type\_name* IS REF CURSOR [ RETURN *return\_type* ];**

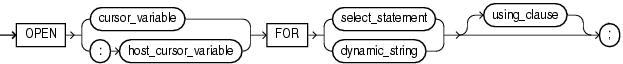


**cursor\_variable type\_name;**

cursor_variable_declaration.gif

### Открытие и закрытие

**OPEN cursor\_variable FOR select\_statement;**



Открытие курсорной переменной выполняет те же действия, что и в случае explicit cursor. Разве что в начале связывает курсорную переменную с конкретным запросом (который может содержать bind переменные)

1. Выделяет ресурсы базы данных для выполнения запроса
2. Выполняет запрос
3. Устанавливает позицию курсора перед первой записью результирующего набора

**CLOSE cursor\_variable;**

Необязательно закрывать курсор перед его переоткрытием с другим запросом. Если закрыть курсор, его можно открыть заново (с последний связанным запросом).

### Извлечение данных

FETCH

### Связывание значения с курсорной переменной

*target\_cursor\_variable* := *source\_cursor\_variable*;

**Исключения:**

* ROWTYPE\_MISMATCH

**Атрибуты у курсорной переменной те же, что и у explicit cursor.**

**Пример:**

**DECLARE**

**TYPE empcurtyp IS REF CURSOR RETURN employees%ROWTYPE; *-- strong type***

**TYPE genericcurtyp IS REF CURSOR; *-- weak type***

**cursor1 empcurtyp; *-- strong cursor variable***

**cursor2 genericcurtyp; *-- weak cursor variable***

**my\_cursor SYS\_REFCURSOR; *-- weak cursor variable***

**v\_employees employees%ROWTYPE;**

**TYPE emplist IS TABLE OF employees%ROWTYPE;**

**emp\_list emplist;**

**v\_sql varchar2(200) := 'SELECT \* FROM employees WHERE REGEXP\_LIKE(job\_id, ''S[HT]\_CLERK'') ORDER BY last\_name';**

**BEGIN**

**OPEN cursor1 FOR v\_sql;**

**LOOP**

**FETCH cursor1**

**INTO v\_employees;**

**EXIT WHEN cursor1%NOTFOUND;**

**some\_actions;**

**END LOOP;**

**CLOSE cursor1;**

**OPEN cursor1 FOR v\_sql;**

**FETCH cursor1 BULK COLLECT**

**INTO emp\_list;**

**CLOSE cursor1;**

**some\_actions\_with\_emp\_list;**

**END;**

# Cursor expression

* Возвращает вложенный курсор:

**CURSOR ( *subquery* )**

* Открывается неявно при извлечении
* Закрывается либо явно пользователем, либо при закрытии родительского курсора

**Пример**:

**DECLARE**

**TYPE emp\_cur\_typ IS REF CURSOR;**

**emp\_cur emp\_cur\_typ;**

**dept\_name departments.department\_name%TYPE;**

**emp\_name employees.last\_name%TYPE;**

**CURSOR c1 IS**

**SELECT department\_name,**

**CURSOR ( SELECT e.last\_name**

**FROM employees e**

**WHERE e.department\_id = d.department\_id**

**ORDER BY e.last\_name**

**) employees**

**FROM departments d**

**WHERE department\_name LIKE 'A%'**

**ORDER BY department\_name;**

**BEGIN**

**OPEN c1;**

**LOOP *-- Process each row of query result set***

**FETCH c1 INTO dept\_name, emp\_cur;**

**EXIT WHEN c1%NOTFOUND;**

**DBMS\_OUTPUT.PUT\_LINE('Department: ' || dept\_name);**

**LOOP *-- Process each row of subquery result set***

**FETCH emp\_cur INTO emp\_name;**

**EXIT WHEN emp\_cur%NOTFOUND;**

**DBMS\_OUTPUT.PUT\_LINE('-- Employee: ' || emp\_name);**

**END LOOP;**

**END LOOP;**

**CLOSE c1;**

**END;**

# Параметры БД

* **select \* from v$parameter where name like '%cursor%‘**
* **select \* from v$system\_parameter where name like '%cursor%‘**

|  |  |
| --- | --- |
| **Параметр** | **Описание** |
| **open\_cursors** | Максимальное количество открытых курсоров в рамках одной сессии (по умолчанию 50). Максимально возможное значение 65 535. |
| **session\_cached\_cursors** | Максимальное количество session cursors в кэше (по умолчанию 50). Курсоры попадают в кэш в случае повторного разбора (parse) одного и того же SQL (включая рекурсивный) – при этом из кэша могут удаляться курсоры, к которым давно не было обращений. |
| **cursor\_space\_for\_time** | Параметр устаревший. Пользоваться не следует. Раньше задавал, следует ли при необходимости увеличивать размер выделяемой памяти для курсора, чтобы уменьшить cpu time. |

# Словари данных

* Представление V$OPEN\_CURSOR возвращает список кэшированных сессией курсоров
* Чтобы получить количество открытых курсоров, используйте запрос:

**SELECT a.value,**

**s.username,**

**s.sid,**

**s.serial#**

**FROM v$sesstat a,**

**v$statname b,**

**v$session s**

**WHERE a.statistic# = b.statistic#**

**AND s.sid = a.sid**

**AND b.name = 'opened cursors current'**