Управляющие конструкции. Операторы цикла Basic Loop, While, For. Вложенные циклы.

**Basic Loop – самый простой вид LOOP-выражений**

Начинается с LOOP и заканчивается END LOOP.

Используется, когда код внутри цикла должен исполниться хотя бы 1 раз.

Синтаксис:

BEGIN

LOOP

*Statements*

EXIT [WHEN condition]

END LOOP;

END;

Простой пример:

-увеличение счетчика на один каждый раз

DECLARE

v\_counter NUMBER(2) :=1;

BEGIN

LOOP

DBMS\_OUTPUT.PUT\_LINE(‘Loop execution #’ || v\_counter);

V\_counter := v\_counter +1;

EXIT WHEN v\_counter>5;

END LOOP;

END;

Изображение выглядит как снимок экрана

Автоматически созданное описание

Сложный пример:

-мы вносим в таблицу LOCATIONS три новые location IDs для Montreal Canada

DECLARE

v\_loc\_id locations.location%TYPE;

v\_counter NUMBER(2) := 1;

BEGIN

SELECT MAX(locayion\_id) INTO v\_loc\_id FROM locations

WHERE country id =2;

LOOP

INSERT INTO locations(location\_id, city, country\_id)

VALUES((v\_loc\_id+v\_counter),’Montreal’,2);

V\_counter := v\_counter +1;

EXIT WHEN v\_counter>3;

END LOOP;

END;

* Можно использовать EXIT, чтобы завершить цикл и перейти к следующему действию сразу после END утверждения.
* Можно использовать EXIT вместе с IF выражением. Пример:

DECLARE

v\_counter NUMBER(2) :=1;

BEGIN

LOOP

DBMS\_OUTPUT.PUT\_LINE(‘Counter is ’ || v\_counter);

V\_counter := v\_counter +1;

IF v\_counter >10 then EXIT;

END IF;

END LOOP;

END;

Изображение выглядит как снимок экрана

Автоматически созданное описание

Правила для EXIT:

* Ставится внутри LOOP
* Если EXIT расположен в начале выполнения цикла перед всеми остальными операциями и истин, то LOOP существует, но ничего другого внутри него после EXIT никогда не выполняется
* BASIC LOOP может содержать в себе несколько EXIT

EXIT WHEN строчка:

* Несмотря на то, что выражение IF … THEN EXIT заканчивает LOOP, наиболее правильным способом закончить LOOP является фраза EXIT WHEN.
* Если EXIT WHEN утверждение является истинным, то LOOP заканчивает свое выполнение и передает очередь следующим операциям после END LOOP

Словарь терминов:

* Basic Loop
* Counter – счетчик
* END LOOP
* EXIT
* LOOP

**WHILE и FOR LOOPS**

1. **While** используется, когда наблюдаемое состояние для выполнения сравнивается с условием каждый раз в начале цикла ( выполняется до тех пор, пока утверждение в начале цикла более не истинно, то есть неизвестно сколько раз будет повторяться цикл)

* Цикл заканчивается при FALSE или NULL
* Если утверждение изначально False или null, то цикл не выполнится ни разу

WHILE condition LOOP

Statement1;

Statement2;

…

END LOOP;

* Condition – булевская переменная (TRUE,FALSE, NULL)
* Может быть несколько statement внутри WHILE LOOP
* Если переменные обозначенные в condition не меняются в теле цикла, то condition всегда TRUE и получается зацикливание.
* Если condition NULL

Пример:

-мы вносим в таблицу LOCATIONS три новые location IDs для Montreal Canada

DECLARE

v\_loc\_id locations.location%TYPE;

v\_counter NUMBER(2) := 1;

BEGIN

SELECT MAX(locayion\_id) INTO v\_loc\_id FROM locations

WHERE country id =2;

WHILE v\_counter<=3 LOOP

INSERT INTO locations(location\_id, city, country\_id)

VALUES((v\_loc\_id+v\_counter),’Montreal’,2);

V\_counter := v\_counter +1;

END LOOP;

END;

* Счетчик явно определен в начале
* После каждого выполнения LOOP v\_counter увеличивается
* Цикл выполняется до тех пор, пока счетчик меньше или равен 3
* Затем после 3 в значении счетчика условие LOOP становится FALSE и цикл завершается.

1. **FOR** используется, когда известно количество выполнения операции цикла

* Почти то же самое, что и обычные LOOP, только в начале LOOP устанавливается количество итераций

FOR counter IN [REVERSE]

Lower\_bound..upper\_bound LOOP

Statement1;

Statement2;

…

END LOOP;

* Счетчик устанавливается неявно
* Lower\_bound..upper\_bound – необходимая часть (нижнее число и верхнее число границы)
* Если используется REVERSE, то счетчик уменьшается ( например, при границе от 9 до 3) (изменение идет на 1)
* Lower\_bound..upper\_bound не должны быть NULL

Пример:

-мы вносим в таблицу LOCATIONS три новые location IDs для Montreal Canada

DECLARE

V\_loc\_id locations.location\_id%TYPE

BEGIN

SELECT MAX(location\_id) INTO v\_Loc\_id FROM locations

WHERE country\_id = 2;

FOR i IN 1..3 LOOP

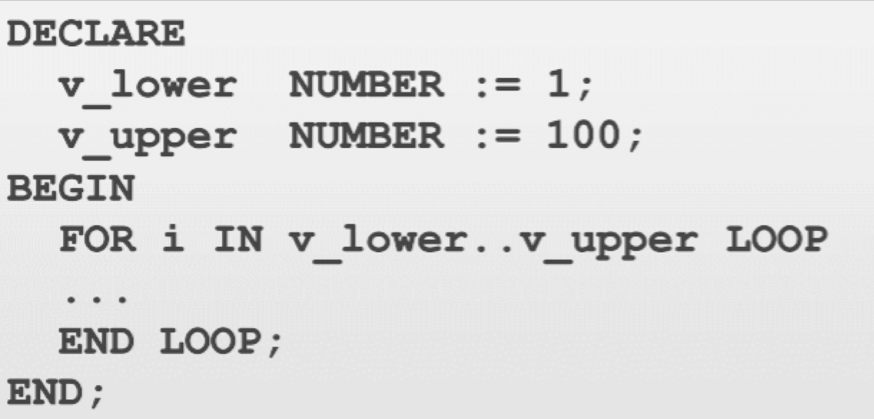
INSERT INTO locations(location\_id, city, country\_id)

VALUES((v\_loc\_id+v\_counter),’Montreal’,2);

END LOOP;

END;

* Lower\_bound..upper\_bound могут быть не только цифровыми значениями
* Пример:



Когда что использовать:

* BASIC LOOP, когда утверждение внутри должно выполниться хотя бы раз
* WHILE: условие должно оцениваться в начале каждого выполнения цикла
* FOR: известно количество итераций

**Вложенные циклы**

* Можно вкладывать циклы FOR,WHILE и basic loop друг в друга

Пример:

BEGIN

FOR v\_outerloop IN 1..3 LOOP

FOR v\_innerloop IN REVERSE 1..500 LOOP

DBMS\_OUTPUT.PUT\_LINE (‘Outer loop is:’ || v\_outerloop || ‘ and inner loop is ‘ || v\_innerloop);

END LOOP;

END LOOP;

END;

Изображение выглядит как снимок экрана

Автоматически созданное описание

LOOP LABELS: нужны для того, чтобы не запутаться в циклах и выйти из внешнего цикла внутри вложенного.

* Название label заключается в <<name>> кавычки
* Если имя label существует, то ставится перед началом цикла на отдельной строчке
* Для большей ясности название label без кавычек ставится в конце цикла вместе с END LOOP
* Изображение выглядит как снимок экрана

  Автоматически созданное описание

Изображение выглядит как снимок экрана

Автоматически созданное описание

Пример:

Изображение выглядит как снимок экрана

Автоматически созданное описание

- два цикла outer\_loop – внешний, inner\_loop - вложенный

- первый EXIT при истинности условия завершит оба цикла

- второй EXIT при истинности завершит внутренний цикл