PL/SQL

Control structures

DE HOGESCHOOL MET HET NETWERK

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be

Controlling Flow of Execution

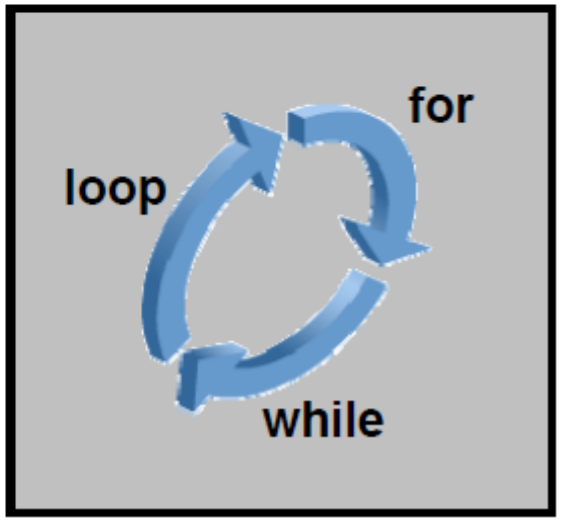
```
IF...
  THEN...
END IF;
```

```
IF...
  THEN...
  ELSE...
END IF;
```

```
IF...
  THEN...
  ELSIF...
  THEN...
END IF;
```

```
IF...
  THEN...
  ELSIF...
  THEN...
  ELSE...
END IF;
```

```
CASE
  WHEN... THEN..
  WHEN... THEN..
  WHEN... THEN..
  ELSE
END CASE;
```



IF Statement

Syntax:

```
IF condition THEN
  statements;
[ELSIF condition THEN
  statements;]
[ELSE
  statements;]
END IF;
```

IF THEN ELSE Statement

```
DECLARE
v_myage  number:=31;
BEGIN
IF v_myage  < 11
 THEN
    DBMS_OUTPUT.PUT_LINE(' I am a child ');
 ELSE
    DBMS_OUTPUT.PUT_LINE(' I am not a child ');
END IF;
END;
/
```

```
anonymous block completed
 I am not a child
```

IF ELSIF ELSE Clause

```
DECLARE
  v_myage number:=31;
BEGIN
  IF v_myage  < 11 THEN
       DBMS_OUTPUT.PUT_LINE(' I am a child ');
    ELSIF v_myage  < 20 THEN
       DBMS_OUTPUT.PUT_LINE(' I am young ');
    ELSIF v_myage  < 30 THEN
       DBMS_OUTPUT.PUT_LINE(' I am in my twenties');
    ELSIF v_myage  < 40 THEN
       DBMS_OUTPUT.PUT_LINE(' I am in my thirties');
    ELSE
       DBMS_OUTPUT.PUT_LINE(' I am always young ');
  END IF;
END;
/
```

```
anonymous block completed
 I am in my thirties
```

NULL Value in IF Statement

```
DECLARE
  v_myage  number;
BEGIN
  IF v_myage  < 11 THEN
    DBMS_OUTPUT.PUT_LINE(' I am a child ');
  ELSE
    DBMS_OUTPUT.PUT_LINE(' I am not a child ');
  END IF;
END;
/
```

```
anonymous block completed
 I am not a child
```

Iterative Control: `LOOP` Statements

- Loops repeat a statement (or sequence of statements) multiple times.
- There are three loop types:
  - Basic loop
  - `FOR` loop
  - `WHILE` loop

Basic Loops

Syntax:

```
LOOP
  statement1;
  . . .
  EXIT [WHEN condition];
END LOOP;
```

Opmerking: herhaal tot …

Basic Loops

Example:

```
DECLARE
  v_countryid    locations.country_id%TYPE := 'CA';
  v_loc_id       locations.location_id%TYPE;
  v_counter      NUMBER(2) := 1;
  v_new_city     locations.city%TYPE := 'Montreal';
BEGIN
  SELECT MAX(location_id) INTO v_loc_id FROM locations
  WHERE country_id = v_countryid;
  LOOP
    INSERT INTO locations(location_id, city, country_id)
    VALUES((v_loc_id + v_counter), v_new_city, v_countryid);
    v_counter := v_counter + 1;
    EXIT WHEN v_counter > 3;
  END LOOP;
END;
/
```

WHILE Loops

Syntax:

```
WHILE condition LOOP
  statement1;
  statement2;
  . . .
END LOOP;
```

Use the WHILE loop to repeat statements while a condition is TRUE.

Opmerking: herhaal zolang …

WHILE Loops: Example

```
DECLARE
  v_countryid   locations.country_id%TYPE := 'CA';
  v_loc_id      locations.location_id%TYPE;
  v_new_city    locations.city%TYPE := 'Montreal';
  v_counter     NUMBER := 1;
BEGIN
  SELECT MAX(location_id) INTO v_loc_id FROM locations
  WHERE country_id = v_countryid;
  WHILE v_counter <= 3 LOOP
    INSERT INTO locations(location_id, city, country_id)
    VALUES((v_loc_id + v_counter), v_new_city, v_countryid);
    v_counter := v_counter + 1;
  END LOOP;
END;
/
```

FOR Loops

- Use a FOR loop to shortcut the test for the number of iterations.
- Do not declare the counter; it is declared implicitly.

```
FOR counter IN [REVERSE]
    lower_bound..upper_bound LOOP
  statement1;
  statement2;
  . . .
END LOOP;
```

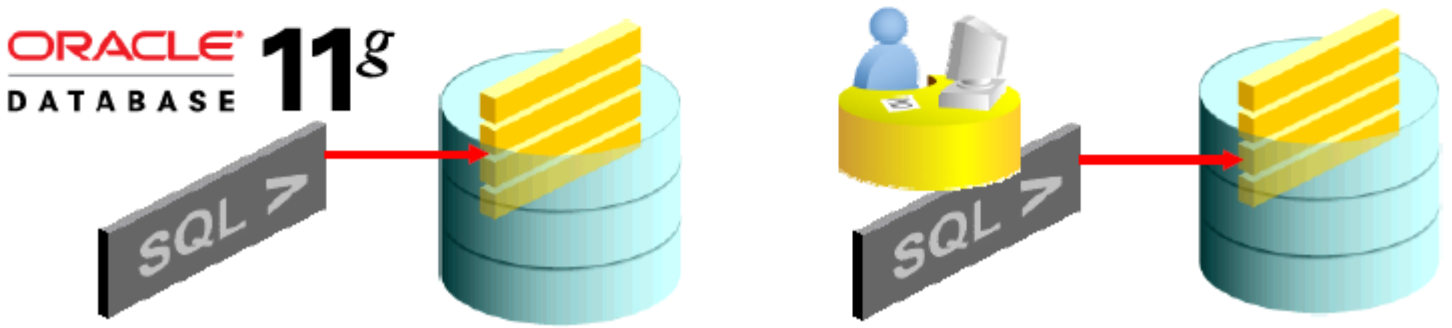
Opmerking: zelftellende lus
teller i wordt impliciet gedeclareerd en kan enkel in lus worden gebruikt – kan wel toegewezen worden aan een variabele en deze is bruikbaar buiten de lus

FOR Loops: Example

```
DECLARE
  v_countryid   locations.country_id%TYPE := 'CA';
  v_loc_id      locations.location_id%TYPE;
  v_new_city    locations.city%TYPE := 'Montreal';
BEGIN
  SELECT MAX(location_id) INTO v_loc_id
    FROM locations
    WHERE country_id = v_countryid;
  FOR i IN 1..3 LOOP
    INSERT INTO locations(location_id, city, country_id)
    VALUES((v_loc_id + i), v_new_city, v_countryid );
  END LOOP;
END;
/
```

SQL Cursor

- A cursor is a pointer to the private memory area allocated by the Oracle server. It is used to handle the result set of a `SELECT` statement.
- There are two types of cursors: Implicit and explicit.
  - **Implicit:** Created and managed internally by the Oracle server to process SQL statements
  - **Explicit:** Declared explicitly by the programmer



Implicit cursor Explicit cursor

SQL Cursor Attributes for Implicit Cursors

Using SQL cursor attributes, you can test the outcome of your SQL statements.

`SQL%FOUND`	Boolean attribute that evaluates to `TRUE` if the most recent SQL statement returned at least one row
`SQL%NOTFOUND`	Boolean attribute that evaluates to `TRUE` if the most recent SQL statement did not return even one row
`SQL%ROWCOUNT`	An integer value that represents the number of rows affected by the most recent SQL statement

SQL Cursor Attributes for Implicit Cursors

Delete rows that have the specified employee ID from the `employees` table. Print the number of rows deleted.

Example:

```
DECLARE
  v_rows_deleted VARCHAR2(30)
  v_empno employees.employee_id%TYPE := 176;
BEGIN
  DELETE FROM  employees
  WHERE employee_id = v_empno;
  v_rows_deleted := (SQL%ROWCOUNT ||
                        ' row deleted.');
  DBMS_OUTPUT.PUT_LINE (v_rows_deleted);

END;
```

Cursor FOR Loops Using Subqueries

There is no need to declare the cursor.

```
BEGIN
  FOR emp_record IN (SELECT employee_id, last_name
   FROM employees WHERE department_id =30)
  LOOP
   DBMS_OUTPUT.PUT_LINE( emp_record.employee_id
   ||' '||emp_record.last_name);
  END LOOP;
END;
/
```

```
anonymous block completed
114 Raphaely
115 Khoo
116 Baida
117 Tobias
118 Himuro
119 Colmenares
```

Cursor FOR Loops Using Subqueries

- Alle rijen en kolommen bekomen door het uitvoeren van de subquery worden in een expliciete cursor bijgehouden
- Deze cursor heeft geen naam en daarom kan er ook geen gebruik gemaakt worden van cursorattributen
  => rec%rowcount kan NIET GEBRUIKT worden
- De FOR-loop zal rij per rij verwerken
- In de LOOP kan er verwezen worden naar een specifiek attribuut via rec.department_name – het gaat hier dan over de inhoud van het attribuut department_name in de rij die op dat moment door de loop verwerkt wordt