

Database administration advanced

Andres Martínez Gutiérrez. 2018-1

Agenda

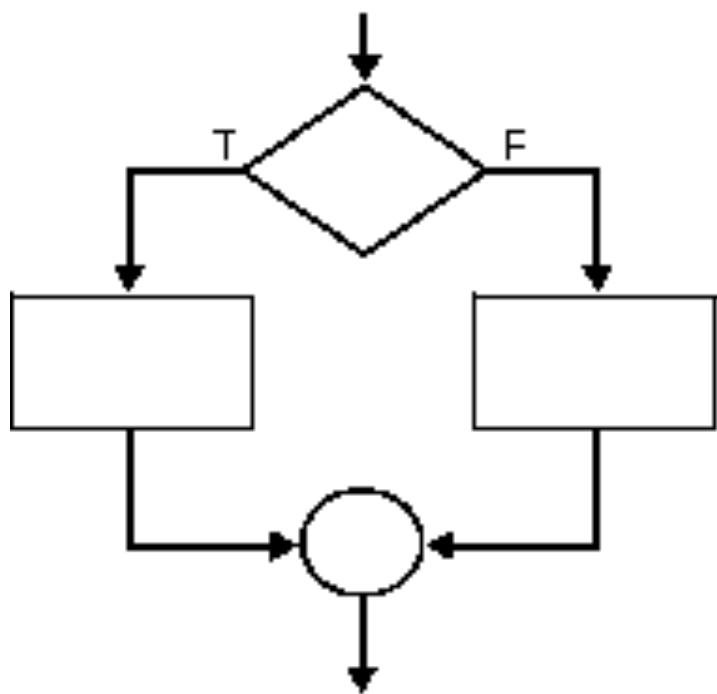
1. PL/SQL Control Structures



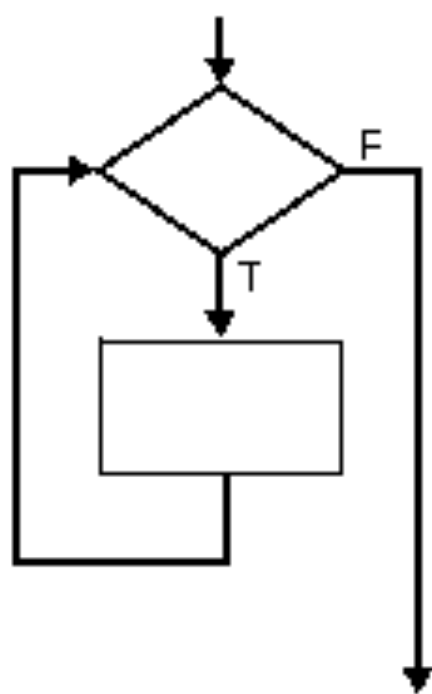
PL/SQL Control Structures

https://docs.oracle.com/cd/A97630_01/appdev.920/a96624/04_struc.htm

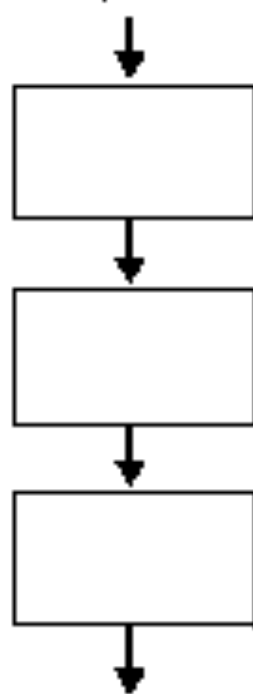
Selection



Iteration



Sequence



Conditional Control: IF and CASE Statements

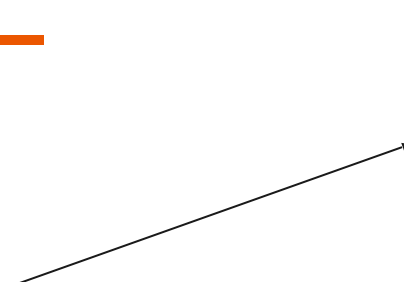
```
1  IF condition THEN
2      sequence_of_statements
3  END IF;
4
5  IF sales > quota THEN
6      compute_bonus(empid);
7      UPDATE payroll SET pay = pay + bonus WHERE empno = emp_id;
8  END IF;
9
10 IF condition THEN
11     sequence_of_statements1
12 ELSE
13     sequence_of_statements2
14 END IF;
15
16 IF trans_type = 'CR' THEN
17     UPDATE accounts SET balance = balance + credit WHERE ...
18 ELSE
19     UPDATE accounts SET balance = balance - debit WHERE ...
20 END IF;
```

Conditional Control: IF-THEN-ELSIF Statement



```
1 IF condition1 THEN
2     sequence_of_statements1
3 ELSIF condition2 THEN
4     sequence_of_statements2
5 ELSE
6     sequence_of_statements3
7 END IF;
8
9 BEGIN
10     ...
11     IF sales > 50000 THEN
12         bonus := 1500;
13     ELSIF sales > 35000 THEN
14         bonus := 500;
15     ELSE
16         bonus := 100;
17     END IF;
18     INSERT INTO payroll VALUES (emp_id, bonus, ...);
19 END;
```

Conditional Control: CASE Statement



The diagram illustrates the equivalence between a CASE statement and an IF-ELSIF statement. A horizontal bar with a teal left half and an orange right half is positioned above the CASE statement. An arrow points from this bar to the IF-ELSIF statement below. The CASE statement (lines 15-22) uses 'WHEN' clauses to check for grades 'A' through 'F', with an 'ELSE' clause for unknown grades. The IF-ELSIF statement (lines 1-13) uses 'IF' and 'ELSIF' clauses to perform the same checks and output the corresponding grade description.

```
15 CASE grade
16     WHEN 'A' THEN dbms_output.put_line('Excellent');
17     WHEN 'B' THEN dbms_output.put_line('Very Good');
18     WHEN 'C' THEN dbms_output.put_line('Good');
19     WHEN 'D' THEN dbms_output.put_line('Fair');
20     WHEN 'F' THEN dbms_output.put_line('Poor');
21     ELSE dbms_output.put_line('No such grade');
22 END CASE;

1 IF grade = 'A' THEN
2     dbms_output.put_line('Excellent');
3 ELSIF grade = 'B' THEN
4     dbms_output.put_line('Very Good');
5 ELSIF grade = 'C' THEN
6     dbms_output.put_line('Good');
7 ELSIF grade = 'D' THEN
8     dbms_output.put_line('Fair');
9 ELSIF grade = 'F' THEN
10    dbms_output.put_line('Poor');
11 ELSE
12    dbms_output.put_line('No such grade');
13 END IF;
```

Conditional Control: CASE Statement

```
1 CASE
2   WHEN grade = 'A' THEN dbms_output.put_line('Excellent');
3   WHEN grade = 'B' THEN dbms_output.put_line('Very Good');
4   WHEN grade = 'C' THEN dbms_output.put_line('Good');
5   WHEN grade = 'D' THEN dbms_output.put_line('Fair');
6   WHEN grade = 'F' THEN dbms_output.put_line('Poor');
7   ELSE dbms_output.put_line('No such grade');
8 END CASE;
```


Iterative Control: LOOP and EXIT Statements

```
1  LOOP
2      sequence_of_statements
3  END LOOP;
4
5  LOOP
6      ...
7  IF credit_rating < 3 THEN
8      ...
9      EXIT;  -- exit loop immediately
10 END IF;
11 END LOOP;
12 -- control resumes here
```


Iterative Control: EXIT-WHEN



```
1 IF count > 100 THEN  
2   EXIT;  
3 END IF;
```


```
EXIT WHEN count > 100;
```

Iterative Control: Loop Labels



```
1  <<my_loop>>
2  LOOP
3      ...
4  END LOOP my_loop;
5
6  <<outer>>
7  LOOP
8      ...
9      LOOP
10         ...
11         EXIT outer WHEN ...  -- exit both loops
12     END LOOP;
13     ...
14 END LOOP outer;
```

Iterative Control: WHILE-LOOP




```
1 WHILE total <= 25000 LOOP
2     ...
3     SELECT sal INTO salary FROM emp WHERE ...
4     total := total + salary;
5 END LOOP;
6
7 LOOP
8     sequence_of_statements
9     EXIT WHEN boolean_expression;
10 END LOOP;
11
12 done := FALSE;
13 WHILE NOT done LOOP
14     sequence_of_statements
15     done := boolean_expression;
16 END LOOP;
```

Iterative Control: FOR-LOOP

```
1  FOR i IN 1..3 LOOP  -- assign the values 1,2,3 to i
2  |   sequence_of_statements  -- executes three times
3  END LOOP;
4
5  FOR i IN 3..3 LOOP  -- assign the value 3 to i
6  |   sequence_of_statements  -- executes one time
7  END LOOP;
8
9  FOR i IN REVERSE 1..3 LOOP  -- assign the values 3,2,1 to i
10 |   sequence_of_statements  -- executes three times
11 END LOOP;
```

NULL Statement



```
1 EXCEPTION
2   WHEN ZERO_DIVIDE THEN
3     ROLLBACK;
4   WHEN VALUE_ERROR THEN
5     INSERT INTO errors VALUES ...
6     COMMIT;
7   WHEN OTHERS THEN
8     NULL;
9 END;
10
11 IF rating > 90 THEN
12   compute_bonus(emp_id);
13 ELSE
14   NULL;
15 END IF;
```



Thank you.