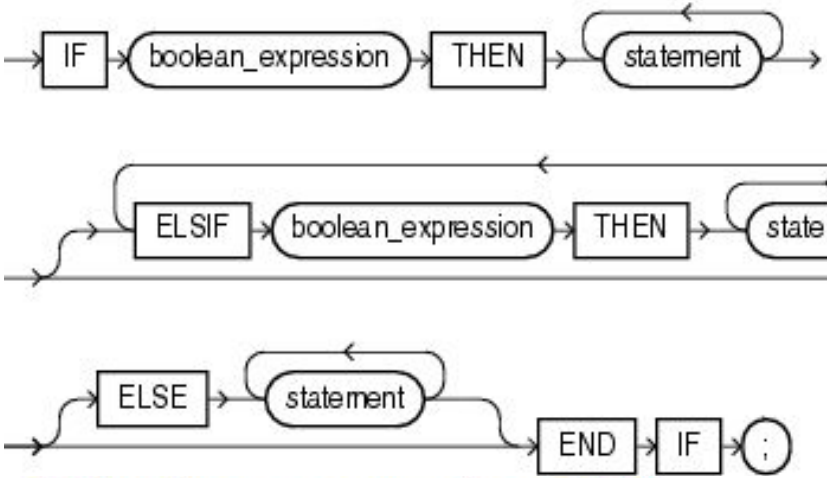
STRUCTURE DE CONTRÔLE : IF - THEN - ELSE




```
IF boolean_expression THEN
    statement [statement]...
 [ELSIF boolean_expression THEN
    statement [statement]...]
   [ELSIF boolean_expression THEN
   statement [statement]...]...
 [ELSE statement [statement]...]
END IF;
```

EXERCICE

/*
EXERCICE 1: Demander à l'utilisateur de saisir un nombre et lui dire si ce nombre est SUP ou INF à 100

EXERCICE 2:
Demandez à l'utilisateur de saisir l'id d'un employé
si :
- le job_id = **IT_PROG** alors **afficher** son salaire + une augmentation de 0.8%
- le job_id = **FI_ACCOUNT** alors **afficher** son salaire + une augmentation de 0.5%
- le job_id = **ST_CLERK** alors **afficher** son salaire + une augmentation de 0.3%
- si non 0.2%
*/

EXERCICE preparation

```
select * from employees;
desc employees;
select job_id from employees;
select distinct job_id from employees
where job_id in ('AC_ACCOUNT','FI_ACCOUNT','FI_MGR');
---
select  EMPLOYEE_ID from employees
where job_id in ('AC_ACCOUNT','FI_ACCOUNT','FI_MGR');
-- Augmentation de salaire 0.9 pour AC_ACCOUNT
-- 0.08 pour  FI_ACCOUNT
-- 0.07 pour FI_MGR
```

EXERCICE preparation

```
DECLARE
   EMPID      EMPLOYEES.EMPLOYEE_ID%TYPE ;
      JOBID      EMPLOYEES.JOB_ID%TYPE ;
      ENAME EMPLOYEES.FIRST_NAME%TYPE;
   SAL_RAISE  NUMBER(3,2);
BEGIN
 EMPID := '&empid';
  DBMS_OUTPUT.PUT_LINE ('pour l''employé :'||EMPID);
  SELECT JOB_ID,FIRST_NAME INTO JOBID,ENAME FROM EMPLOYEES WHERE EMPLOYEE_ID = EMPID;
  IF JOBID = 'AC_ACCOUNT' THEN SAL_RAISE := .09;
  ELSIF JOBID = 'FI_ACCOUNT' THEN SAL_RAISE := .08;
  ELSIF JOBID = 'FI_MGR' THEN SAL_RAISE := .07;
  ELSE SAL_RAISE := 0;
  END IF;
  DBMS_OUTPUT.PUT_LINE (ENAME||' sera augmenté de  '||SAL_RAISE||' job id'||JOBID);
END;
```

IF THEN ELSE

```
DECLARE
v_num NUMBER := &sv_user_num;
BEGIN
    IF MOD(v_num,2) = 0 THEN
        DBMS_OUTPUT.PUT_LINE (v_num||' est un nombre paire');
    ELSE
        DBMS_OUTPUT.PUT_LINE (v_num||' un nombre impaire');
    END IF;
DBMS_OUTPUT.PUT_LINE (' Fin');
END;
```

IF THEN ELSE

```
SET SERVEROUTPUT ON
DECLARE
v_date DATE := TO_DATE('&sv_user_date', 'DD/MM/YYYY');
v_day VARCHAR2(15);
BEGIN
v_day := TO_CHAR(v_date, 'DAY');
DBMS_OUTPUT.PUT_LINE (' ce sera un :'||v_day);
IF v_day IN ('SAMEDI', 'DIMANCHE') THEN
DBMS_OUTPUT.PUT_LINE (v_day||', week end!');
END IF;
-- control resumes here
DBMS_OUTPUT.PUT_LINE ('Done...');
END;
```

UTILISATION DU ELSIF

```
DECLARE
v_num NUMBER := &sv_num;
BEGIN
IF v_num < 0 THEN
DBMS_OUTPUT.PUT_LINE (v_num||' nombre négatif');
ELSIF v_num > 0 THEN
DBMS_OUTPUT.PUT_LINE (v_num||' nombre positif');
END IF;
DBMS_OUTPUT.PUT_LINE ('Done...');
END;
```

CASE NULLIF ET COALESCE

```
DECLARE
v_num NUMBER := &sv_user_num;
v_num_flag NUMBER;
BEGIN
v_num_flag := MOD(v_num,2);
CASE v_num_flag
WHEN 0 THEN DBMS_OUTPUT.PUT_LINE (v_num||' EST PAIRE');
ELSE DBMS_OUTPUT.PUT_LINE (v_num||' EST IMPAIRE');
END CASE;
DBMS_OUTPUT.PUT_LINE ('Done');
END;
```

CASE

```
CASE SELECTOR
   WHEN EXPRESSION 1 THEN STATEMENT 1;
   WHEN EXPRESSION 2 THEN STATEMENT 2;
   ...
   WHEN EXPRESSION N THEN STATEMENT N;
   ELSE STATEMENT N+1;
END CASE;
```

Exercice : Reprendre l'exercice précédent avec CASE

Demander à l'utilisateur de saisir une date,

- ensuite récupérer le jour : **to_char**(date, 'fmt')
- si LUNDI alors afficher 'Au travail'
- si MARDI alors afficher 'Repose'
- si MERCREDI alors afficher 'Pour les enfants'
- si JEUDI alors afficher 'Déjà fatigué'
- si VENDREDI alors afficher 'Vivement le week end'
- autres afficher week-end.

CASE

Exercice : Demander à l'utilisateur de saisir une date,

- ensuite récupérer le numéro du jour grace à la commande : **to_char**(date, 'D')
- et dire s'il s'agit du week-end ou pas.

SWITCH - CASE

```
SET SERVEROUTPUT ON
DECLARE
    v_date DATE := TO_DATE('&sv_user_date', 'DD/MM/YYYY');
    v_day VARCHAR2(1);
BEGIN
    v_day := TO_CHAR(v_date, 'D');
CASE v_day WHEN '1' THEN DBMS_OUTPUT.PUT_LINE ('LUNDI');
    WHEN '2' THEN DBMS_OUTPUT.PUT_LINE ('MARDI');
    WHEN '3' THEN DBMS_OUTPUT.PUT_LINE ('MERCREDI');
    WHEN '4' THEN DBMS_OUTPUT.PUT_LINE ('JEUDI');
    WHEN '5' THEN DBMS_OUTPUT.PUT_LINE ('VENDREDI');
    WHEN '6' THEN DBMS_OUTPUT.PUT_LINE ('SAMEDI');
    WHEN '7' THEN DBMS_OUTPUT.PUT_LINE ('DIMANCHE');
END CASE;
END;
```

NULLIF

A = NULLIF (expression1, expression2)

A sera null si expression1 = expression2,
si non ce sera la valeur de la 1ere expression.

Exercice : Demander à l'utilisateur de saisir un nombre et afficher s'il est pair ou impair.

```
set SERVEROUTPUT ON

declare
v_num number := &u_Num;
v_Reste number;
Begin
  v_reste := nullif(Mod(v_num, 2),0);
  if v_reste is null then
    dbms_output.put_line ('Le reste est null');
  else
    dbms_output.put_line ('Le reste est : '||v_reste);
  end IF;
end;
```

LES BOUCLES

Elles permettent de faire un traitement itératif

```
LOOP
        -- instrcution 1
        -- instrcution 2
        -- instrcution 3
END LOOP
```

LES BOUCLES : EXIT pour finir

```
SET SERVEROUTPUT ON
DECLARE
v_counter BINARY_INTEGER := 0;
BEGIN
    LOOP
        v_counter := v_counter + 1;
        DBMS_OUTPUT.PUT_LINE ('v_counter = '||v_counter);
        IF v_counter = 5 THEN
            EXIT;
        END IF;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE ('Done...');
END;
```

LES BOUCLES

Exercice :
Dans le schéma HR,
créer une table client (id int auto-incrémenté, info varchar2(50))

Insérer 100 enregistrements dans la table client.

(envoyez vos scripts en privé)

LES BOUCLES : EXIT pour finir

```
SET SERVEROUTPUT ON
DECLARE
v_counter BINARY_INTEGER := 0;
BEGIN
LOOP
v_counter := v_counter + 1;
DBMS_OUTPUT.PUT_LINE ('v_counter = '||v_counter);
EXIT WHEN v_counter = 5;
END LOOP;
DBMS_OUTPUT.PUT_LINE ('Done...');
END;
```

LES BOUCLES : while

```
WHILE CONDITION   LOOP
        -- instrcution 1
        -- instrcution 2
        -- instrcution 3
 END LOOP
```

LES BOUCLES : while

```
DECLARE
v_counter NUMBER := 5;
BEGIN
WHILE v_counter < 5 LOOP
DBMS_OUTPUT.PUT_LINE ('v_counter = '||v_counter);
v_counter := v_counter - 1;
END LOOP;
END;
```

LES BOUCLES : while - exit prématuré

```
DECLARE
v_counter NUMBER := 1;
BEGIN
WHILE v_counter <= 5 LOOP
DBMS_OUTPUT.PUT_LINE ('v_counter = '||v_counter);
IF v_counter = 2 THEN
EXIT;
END IF;
v_counter := v_counter + 1;
END LOOP;
END;
```

LES BOUCLES :

Exercice : Donner la sommes des nombres paires et la somme des nombres impairs entre 1 et 100

LES BOUCLES : while - exit prématuré

```
SET SERVEROUTPUT ON
DECLARE
v_counter BINARY_INTEGER := 2;
v_sum NUMBER := 0;
BEGIN
WHILE v_counter <= 100 LOOP
v_sum := v_sum + v_counter;
DBMS_OUTPUT.PUT_LINE ('la somme est : '||v_sum);
-- increment loop counter by two
v_counter := v_counter + 2;
END LOOP;
-- control resumes here
DBMS_OUTPUT.PUT_LINE ('la somme des nombre impairs : '||
'1 et 100 est : '||v_sum);
END;
```

LES BOUCLES - FOR

```
FOR i in v_valeur1 .. v_valeur2
LOOP
  -- instructions ...
END LOOP;
--- autres instructions ...
```

Exercice : Calculer le factoriel de 10

LES BOUCLES : exercice 10!

```
SET SERVEROUTPUT ON
DECLARE
v_factorial NUMBER := 1;
BEGIN
FOR v_counter IN REVERSE 1..10 LOOP
v_factorial := v_factorial * v_counter;
END LOOP;
-- control resumes here
DBMS_OUTPUT.PUT_LINE
('le factoriel de 10 : '||v_factorial);
END;
```

Exercice : En combien de temps votre ordinateur va incrémenter un compteur 900 000 000 000 de fois ?
indication : utiliser DBMS_UTILITY.GET_TIME;

Sequence

Une SEQUENCE un sorte de compteur,utilisé sous Oracle.

Utiliser des séquences pour générer automatiquement des valeurs de clé primaire.

```
CREATE SEQUENCE sequence_name [INCREMENT BY n] [START WITH n] [{MAXVALUE n
| NOMAXVALUE}] [{MINVALUE n | NOMINVALUE}] [{CYCLE | NOCYCLE}] [{CACHE n | NOCACHE}];
```

CREATE SEQUENCE client_id_seq INCREMENT BY 1 START WITH 1001 MAXVALUE 1010 NOCYCLE NOCACHE;
IDENTITY -- l'autre manière de créer des clés primaires auto-incrémenté

LES BOUCLES - FOR

```
--------------------------------------------------------------------
SET SERVEROUTPUT ON
DECLARE
v_valeur1 int := 0;
v_valeur2 int := 1;

v_start number ;
v_stop number ;
BEGIN
v_start := DBMS_UTILITY.GET_TIME;
FOR i in 1..90000000
LOOP
  v_valeur1 := v_valeur1 + v_valeur2;
END LOOP;
    v_stop := DBMS_UTILITY.GET_TIME;
    DBMS_OUTPUT.PUT_LINE ('Départ :'||v_start||' fin :'||v_stop||'. Temps ecoulé: '||
    (v_stop - v_start));
END;
```

Exercice : Créer une séquence et afficher sa valeur

LES BOUCLES - FOR

```
SET SERVEROUTPUT ON
DECLARE
v_seq_value NUMBER;
BEGIN
v_seq_value := test_seq.NEXTVAL;
DBMS_OUTPUT.PUT_LINE ('la valeur de la séquence est : '||v_seq_value);
END;
```

Exercice : Reprendre l'exercice sur temps de traitement des incrémentations et comparer avec l'incrémentation d'une séquence .

LES BOUCLES - FOR

```
SET SERVEROUTPUT ON
DECLARE
v_valeur1 int := 0; v_valeur2 int := 1; v_seq_value number; v_start number ; v_stop number ;
BEGIN
    v_start := DBMS_UTILITY.GET_TIME;
        FOR i in 1..100000
            LOOP
                v_valeur1 := v_valeur1 + v_valeur2;
            END LOOP;
                v_stop := DBMS_UTILITY.GET_TIME;
            DBMS_OUTPUT.PUT_LINE ('Départ :'||v_start||' fin :'||v_stop||'. Temps ecoulé: '||
        (v_stop - v_start));
     v_start := DBMS_UTILITY.GET_TIME;
        FOR i in 1..100000
            LOOP
                v_seq_value := test_seq.NEXTVAL;
            END LOOP;
            v_stop := DBMS_UTILITY.GET_TIME;
            DBMS_OUTPUT.PUT_LINE ('Départ :'||v_start||' fin :'||v_stop||'. Temps ecoulé: '||
            (v_stop - v_start));
      v_start := DBMS_UTILITY.GET_TIME;
        FOR i in 1..10000 LOOP
            SELECT test_seq.NEXTVAL
            INTO v_seq_value
            FROM dual;
        END LOOP;
        v_stop := DBMS_UTILITY.GET_TIME;
     DBMS_OUTPUT.PUT_LINE ('Départ :'||v_start||' fin :'||v_stop||'. Temps ecoulé: '||
    (v_stop - v_start));
    END;
```

SYNTAXE PROCEDURE STOCKEE

```
CREATE [ OR REPLACE]  PROCEDURE Nom_Procedure [( IN/OUT parameter[, parameter, ...])]

AS [variables locales]

BEGIN

      Partie exécutable

[EXCEPTION exception]


 END [Nom_Procedure ];
```

Exemple :
Dans la liste des paramètres on utilise : IN et OUT pour indiquer si les paramètres sont en entrée ou en sortie

OR REPLACE : recréer si existe

LES PROCEDURES STOCKEES : AJOUT

```
CREATE OR REPLACE PROCEDURE Ajout_Client
( p_NOM IN varchar2, p_prenom IN varchar2, p_address IN varchar2, p_cp VARCHAR2, p_tel VARCHAR2  )
as
begin
insert into client
values (client_id_seq.nextval,upper(p_nom), InitCap(p_prenom),p_address,p_cp,p_tel);
end;
begin
Ajout_Client('toto','coco',' fddf ','75000','0101010101');
end;
select * from client;
```

LES PROCEDURES STOCKEES : RECHERCHE

```
CREATE OR REPLACE PROCEDURE find_employee
(p_empid IN NUMBER,name out nvarchar2 )
as
Begin
    select first_name || last_name into name from employees
    where employee_id = p_empid;
end;
SET SERVEROUTPUT ON
declare vName varchar2(50);
begin
  find_employee (100,vName);
 DBMS_OUTPUT.PUT_LINE('Le resultat est '||vName);
end;
```

LES PROCEDURES STOCKEES : RECHERCHE

Exercice : Ecrire une procédure Search_By_ID qui AFFICHE le nom, prenom, telephone du client dont l'id est passé en paramètre si le client existe si non la procédure affiche client inexistant.

Reprendre la procédure Ajout_Client pour vérifier si le numéro de tél est correcte (14 chiffres) et si le code poste est correcte (5 ou 6 caractères)