

**Ecrire des structures
de contrôle**

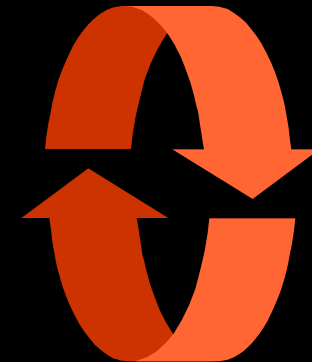
Objectifs

A la fin de ce chapitre, vous pourrez :

- **identifier les types de structure de contrôle et leurs utilisations**
- **écrire une instruction IF**
- **utiliser des expressions CASE**
- **écrire et identifier différents types d'instruction LOOP**
- **utiliser des tables logiques**
- **contrôler le flux de blocs à l'aide de boucles imbriquées et d'étiquettes**

Contrôler le flux d'exécution PL/SQL

- Vous pouvez modifier l'exécution logique des instructions en utilisant des instructions conditionnelles IF et des structures de contrôle LOOP
- Instructions conditionnelles IF :
 - IF-THEN-END IF
 - IF-THEN-ELSE-END IF
 - IF-THEN-ELSIF-END IF



Instructions IF

Syntaxe :

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

Si le nom de l'employé est Gietz, lui affecter l'ID de manager 102.

```
IF UPPER(v_last_name) = 'GIETZ' THEN  
    v_mgr := 102;  
END IF;
```

Instructions IF simples

Si le nom de famille est Vargas :

- **affecter la valeur SA_REP au code de poste**
- **affecter la valeur 80 au numéro du service**

```
. . .  
IF v_ename      = 'Vargas' THEN  
    v_job       := 'SA_REP';  
    v_deptno    := 80;  
END IF;  
. . .
```

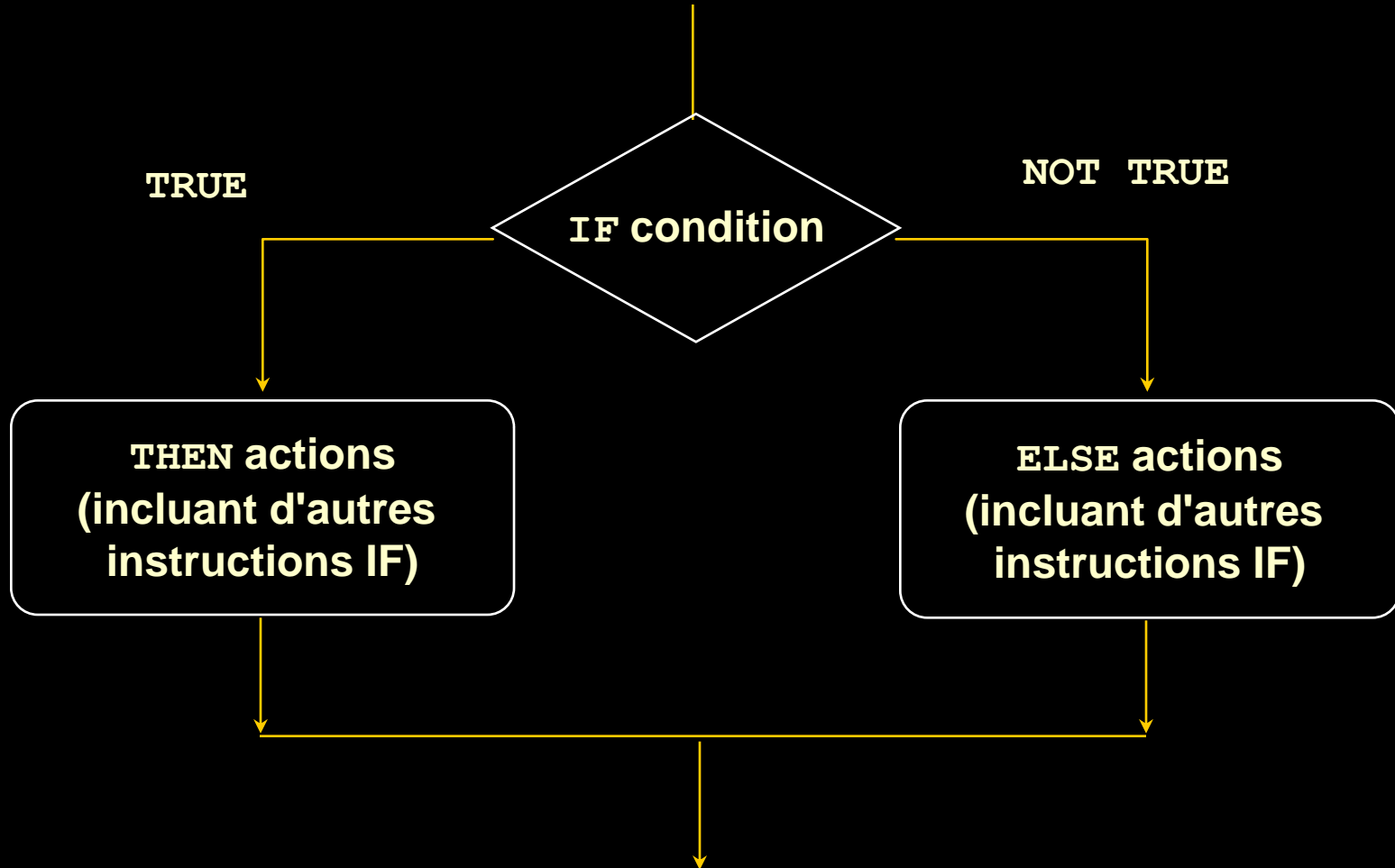
Instructions IF composées

Si le nom de famille est Vargas et que le salaire est supérieur à 6 500 :

affecter la valeur 60 au numéro du service

```
. . .  
IF v_ename = 'Vargas' AND salary > 6500 THEN  
    v_deptno := 60;  
END IF;  
. . .
```

Flux d'exécution des instructions IF-THEN-ELSE



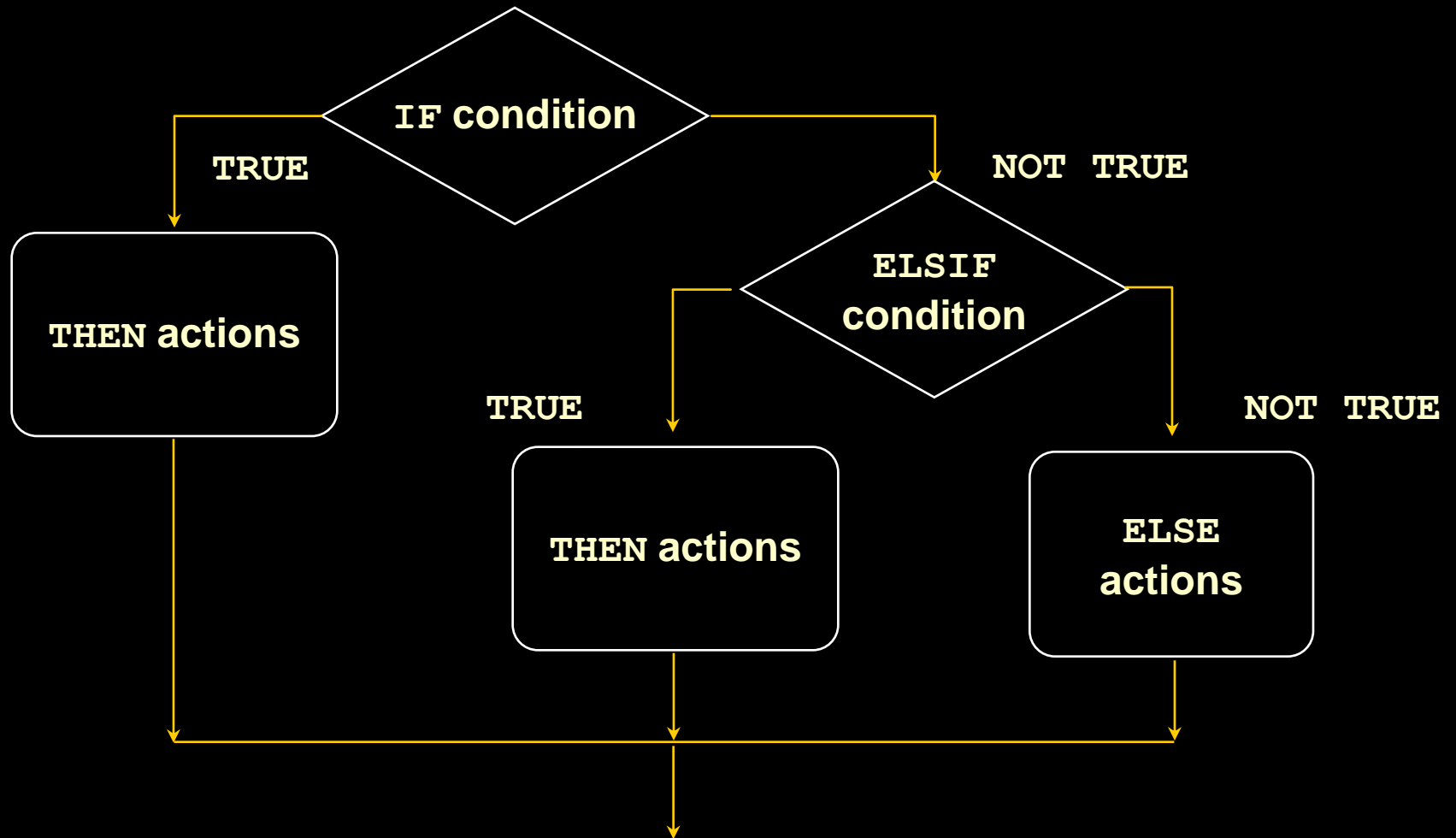
Instructions IF-THEN-ELSE

Affecter la valeur TRUE à un indicateur booléen si la date d'embauche est de plus de cinq ans ; sinon, affecter la valeur FALSE.

```
DECLARE
    v_hire_date  DATE := '12-Dec-1990';
    v_five_years BOOLEAN;
BEGIN
    . . .
    IF MONTHS_BETWEEN(SYSDATE,v_hire_date)/12 > 5 THEN
        v_five_years := TRUE;
    ELSE
        v_five_years := FALSE;
    END IF;
    . . .
```


Flux d'exécution des instructions

IF-THEN-ELSIF



Instructions IF-THEN-ELSIF

Calculer le pourcentage d'une valeur donnée en fonction d'une condition

Exemple :

```
. . .  
IF      v_start > 100 THEN  
        v_start := 0.2 * v_start;  
ELSIF v_start >= 50 THEN  
        v_start := 0.5 * v_start;  
ELSE  
        v_start := 0.1 * v_start;  
END IF;  
. . .
```

Expressions CASE

- Une expression CASE sélectionne un résultat et le renvoie
- Pour que le résultat soit sélectionné, l'expression CASE utilise une expression dont la valeur permet d'effectuer un choix parmi plusieurs possibilités

```
CASE selector
  WHEN expression1 THEN result1
  WHEN expression2 THEN result2
  ...
  WHEN expressionN THEN resultN
[ELSE resultN+1;]
END;
```

Exemple d'expressions CASE

[illegible]

Traiter les valeurs NULL

Lorsque vous utilisez des valeurs NULL, vous pouvez éviter certaines erreurs fréquentes en gardant à l'esprit les règles suivantes :

- les comparaisons simples impliquant des valeurs NULL renvoient toujours une valeur NULL
- l'application de l'opérateur logique NOT à une valeur NULL renvoie une valeur NULL
- dans les instructions de contrôle conditionnelles, si la condition renvoie une valeur NULL, la séquence d'instructions associée n'est pas exécutée

Tables logiques

Créer une condition booléenne simple avec un opérateur de comparaison

AND	<i>TRUE</i>	<i>FALSE</i>	<i>NULL</i>	OR	<i>TRUE</i>	<i>FALSE</i>	<i>NULL</i>	NOT	
<i>TRUE</i>	TRUE	FALSE	NULL	<i>TRUE</i>	TRUE	TRUE	TRUE	<i>TRUE</i>	FALSE
<i>FALSE</i>	FALSE	FALSE	FALSE	<i>FALSE</i>	TRUE	FALSE	NULL	<i>FALSE</i>	TRUE
<i>NULL</i>	NULL	FALSE	NULL	<i>NULL</i>	TRUE	NULL	NULL	<i>NULL</i>	NULL

Conditions booléennes

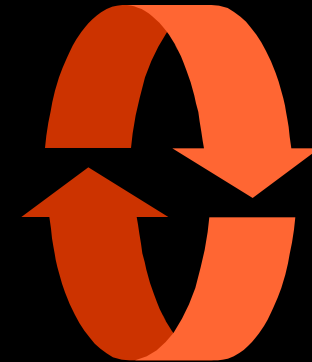
Quelle est la valeur de **V_FLAG** dans chaque cas ?

```
v_flag := v_reorder_flag AND v_available_flag;
```

V_REORDER_FLAG	V_AVAILABLE_FLAG	V_FLAG
TRUE	TRUE	?
TRUE	FALSE	?
NULL	TRUE	?
NULL	FALSE	?

Contrôle d'itération : instructions LOOP

- Les boucles permettent d'exécuter plusieurs fois une instruction ou une séquence d'instructions
- Il existe trois types de boucle :
 - boucle de base
 - boucle FOR
 - boucle WHILE



Boucles de base

Syntaxe :

```
LOOP                                -- delimiter
  statement1;                      -- statements
  . . .
  EXIT [WHEN condition];          -- EXIT statement
END LOOP;                           -- delimiter
```

condition est une variable ou une expression
booléenne (TRUE, FALSE, ou NULL);

Boucles de base


Exemple :

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

Boucles WHILE

Syntaxe :

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



Condition is
evaluated at the
beginning of
each iteration.

Utiliser la boucle **WHILE** pour répéter des instructions tant qu'une condition renvoie **TRUE**.

Boucles WHILE

Exemple :

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

Boucles FOR

Syntaxe :

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

- Utiliser une boucle FOR pour simplifier le contrôle du nombre d'itérations
- Ne pas déclarer le compteur (sa déclaration est implicite)
- La syntaxe requise est '*lower_bound* .. *upper_bound*'

Boucles FOR

Insérer trois nouveaux ID d'emplacement pour le code de pays CA et la ville de Montréal.

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

Boucles FOR

Remarques

- Ne référencer le compteur qu'à l'intérieur de la boucle, il n'est pas défini en dehors
- *Ne pas référencer* le compteur en tant que cible d'une affectation

Remarques relatives à l'utilisation des boucles

- **Utiliser la boucle de base lorsque ses instructions doivent s'exécuter au moins une fois**
- **Utiliser la boucle WHILE si la condition doit être évaluée au début de chaque itération**
- **Utiliser une boucle FOR si le nombre d'itérations est connu**

Boucles imbriquées et étiquettes

- Imbriquer des boucles à plusieurs niveaux
- Utiliser des étiquettes pour différencier les blocs des boucles
- Quitter la boucle externe en utilisant l'instruction `EXIT` qui référence l'étiquette

Boucles imbriquées et étiquettes

```
...  
BEGIN  
  <<Outer_loop>>  
  LOOP  
    v_counter := v_counter+1;  
    EXIT WHEN v_counter>10;  
    <<Inner_loop>>  
    LOOP  
      ...  
      EXIT Outer_loop WHEN total_done = 'YES';  
      -- Leave both loops  
      EXIT WHEN inner_done = 'YES';  
      -- Leave inner loop only  
      ...  
    END LOOP Inner_loop;  
    ...  
  END LOOP Outer_loop;  
END;
```

Synthèse

**Ce chapitre vous a permis d'apprendre à :
modifier l'enchaînement logique des instructions en
utilisant des structures de contrôle**

- **Instructions conditionnelles (IF)**
- **Expressions CASE**
- **Boucles :**
 - **boucle de base**
 - **boucle FOR**
 - **boucle WHILE**
- **Instructions EXIT**

Présentation de l'exercice 4

Dans cet exercice, vous allez :

- **exécuter des actions conditionnelles en utilisant l'instruction IF**
- **écrire des schémas itératifs en utilisant la structure de la boucle**