

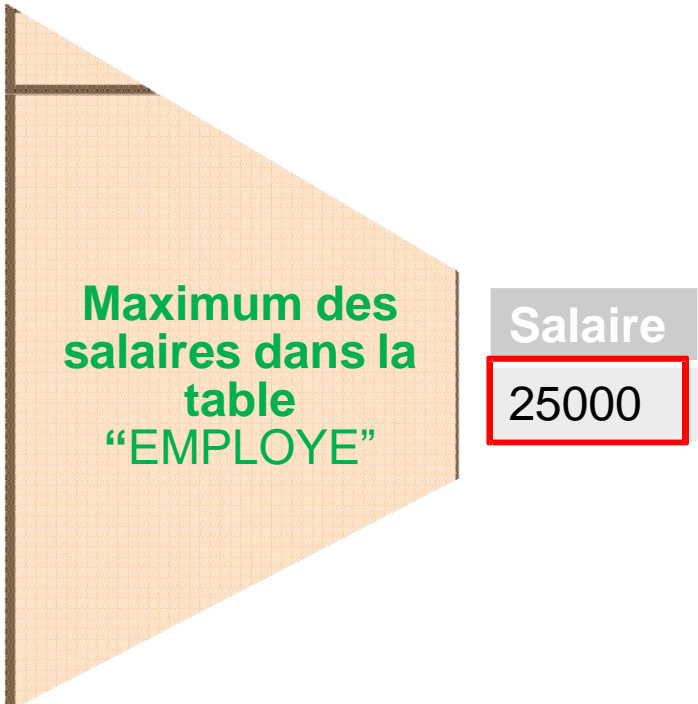
# 4

## Fonctions de groupe

# Fonctions de groupe

- Les fonctions de groupe sont utilisées pour afficher des informations sur **un groupe d'enregistrements**.

Id_employé	Salaire
E20	8000
E21	25000
E22	9400
E23	7000
E24	5000
E25	4900
E26	8100
E27	5900



Maximum des salaires dans la table "EMPLOYE"

Salaire
25000

## Fonctions de groupe (2)

### ❖ Syntaxe:

```
SELECT [colonne,] fonction_groupe(argument)
FROM table
[ WHERE condition(s) ]
[ GROUP BY colonne]
[ORDER BY fonction_group(argument)];
```

### ❖ Remarques:

- L'argument peut-être un nom de colonne, une expression ou une constante.
- Les fonctions de groupe **ne peuvent pas être utilisées** dans les clauses FROM, WHERE et GROUP BY

## Fonctions de groupe (3)

Fonction	Description
<b>SUM</b> ( [DISTINCT   ALL] n )	Retourne la somme de toutes les valeurs du groupe <b>n</b>
<b>MIN</b> ( [DISTINCT   ALL] expr )	Retourne la plus petite valeur du groupe <b>expr</b>
<b>MAX</b> ( [DISTINCT   ALL] expr )	Retourne la plus grande valeur du groupe <b>expr</b>
<b>COUNT</b> ( { *   [DISTINCT   ALL] expr } )	-Retourne le nombre d'enregistrements contenus dans le groupe <b>expr</b> . - <b>COUNT</b> (*) retourne le nombre total d'enregistrements retournés <u>en incluant les valeurs nulles et les doublons.</u>
<b>AVG</b> ( [DISTINCT   ALL] n )	Retourne la moyenne des valeurs du groupe <b>n</b>
<b>STDDEV</b> ( [DISTINCT   ALL] x )	Retourne la déviance standard (l'écart type) de <b>x</b>
<b>VARIANCE</b> ( [DISTINCT   ALL] x )	Retourne la variance de <b>x</b>

# Utilisation des fonctions AVG et SUM

## ❑ Exercice

Ecrire la requête qui retourne la moyenne et la somme des salaires des employés dont la fonction commence par la chaîne de caractère « tech »

# Utilisation des fonctions AVG et SUM (2)

## ❑ Solution

```
SELECT AVG(salaire), SUM(salaire)  
  
FROM employe  
  
WHERE fonction LIKE 'tech%';
```

AVG(salaire)	SUM(salaire)
9162.5	73300

# Utilisation des fonctions MIN et MAX

## ❑ Exercice

Ecrire la requête qui retourne la date d'embauche la plus ancienne et la plus récente de la table **Employé**.

## Fonctions de groupe (4)

### ❖ Remarques:

- Les fonctions de groupe ignorent **les valeurs nulles** sauf la fonction **COUNT(\*)**.
- Le type de données des arguments peuvent être **CHAR**, **VARCHAR2**, **NUMBER**, **DATE** sauf pour les fonctions **AVG**, **SUM**, **VARIANCE** et **STDDEV** qui ne peuvent être utilisées qu'avec **des données de type numérique**.
- Pour substituer les valeurs nulles dans un groupe, il faut utiliser la fonction single-row : « **NVL** »



## Utilisation des fonctions MIN et MAX (2)

### ❑ Exemple:

Afficher le minimum et le maximum des dates  
d'embauche des employés

### ❑ Solution

```
SELECT MIN(date_emb), MAX(date_emb)  
  
FROM employe;
```

MIN(date_emb)	MAX(date_emb)
01/01/1980	01/01/2012

# Utilisation de la fonction COUNT

## ❑ Exercice

- **Q1:** Ecrire la requête qui renvoie le nombre de départements « deptno » (doublons inclus) de la table employe.
- **Q2:** Ecrire la requête qui retourne le nombre d'enregistrements dans la table EMP dont la colonne deptno a pour valeur 30.

**Remarque:** la table EMP est celle présenté dans « Scott ».

## Utilisation de la fonction COUNT (2)

### ❑ Solution Q1

```
SELECT COUNT(deptno)  
FROM employe;
```

COUNT(deptno )
14

### ❑ Solution Q2

```
SELECT COUNT(*)  
FROM employe;  
WHERE deptno = 30;
```

COUNT(*)
6

# Utilisation de la fonction DISTINCT

## ❑ Exercice

- Ecrire la requête qui renvoie le nombre de départements (deptno) en supprimant les doublons de la table employe.

## ❑ Solution

```
SELECT COUNT(DISTINCT(deptno))  
FROM employe;
```

# NVL et fonctions de groupes

## ❑ Exercice

- Ecrire la requête qui renvoie la moyenne des commissions (comm) obtenues par les employés.

## ❑ Solution

```
SELECT AVG(comm)  
FROM employe;
```

4 enregistrements parmi 14 pris en compte: les autres ont la valeur Null dans comm.

AVG(comm)
1000

- **N.B:** Le calcul de la moyenne ne tient pas compte des valeurs invalides telles que les valeurs nulles.

## NVL et fonctions de groupes (2)

### ❑ Exercice

- Ecrire la requête qui renvoie la moyenne des commissions (comm) obtenues par les employés, après le remplacement des valeurs nulles par 0.

### ❑ Solution

```
SELECT AVG(NVL(comm,0))  
FROM employe;
```

14 enregistrements pris en compte,  
c.-à-d. tous les enregistrements

AVG(NVL(comm,0))
157,14

# Création de groupes de données

## ❖ Clause GROUP BY

- Permet de diviser les enregistrements d'une table en groupes.
- Les fonctions de groupe peuvent être utilisées pour retourner les informations relatives à chaque groupe.

## ❖ Syntaxe:

```
SELECT [colonne,] fonction_groupe(argument)
FROM table
[ WHERE condition(s) ]
[ GROUP BY colonne]
[ORDER BY fonction_groupe(argument)];
```

## Création de groupes de données (2)

### ❖ Règles à respecter:

- La clause **WHERE** peut être utilisée pour pré-exclure des enregistrements avant la division en groupes
- Par défaut, la clause **GROUP BY** classe les enregistrements par ordre croissant. L'ordre peut être changé en utilisant la clause **ORDER BY** nom\_col **DESC**.



## Création de groupes de données (4)

### ❑ Exercice

- Ecrire la requête qui renvoie la moyenne des salaires (sal) pour chaque département (deptno) présent dans la table employé.

### ❑ Solution

```
SELECT deptno, AVG(sal) FROM employe GROUP BY deptno;
```

deptno	AVG(sal)
10	2000
20	3500
30	5000

La colonne contenue dans la clause GROUP BY n'a pas obligatoirement besoin de se trouver dans la clause **SELECT**

# Groupes sur plusieurs colonnes

- Plusieurs colonnes peuvent être spécifiées dans la clause GROUP BY
- Récupérer des informations d'un groupe intégré dans un groupe. (Organiser les données en sous-groupe)

## Syntaxe:

```
SELECT [colonne1,colonne2] fonction_groupe(argument)
FROM table
[ WHERE condition(s) ]
[ GROUP BY colonne1, colonne2]
[ORDER BY fonction_group(argument)];
```

- Les données seront organisées en groupes par rapport à la colonne « **colonne1** ». Puis chaque groupe sera à nouveau organisé en sous-groupes par rapport à la colonne « **colonne2** »

## Groupes sur plusieurs colonnes (2)

### ❑ Exercice

- Ecrire la requête qui renvoie la somme des salaires de la table employé pour chaque fonction (fonction), groupé par département (deptno) et fonction.

### ❑ Solution

**SELECT** deptno, fonction, SUM(sal) **FROM** emp **GROUP BY** deptno, fonction;

Deptno	Fonction	SUM(sal)
10	Ingénieur	40000
10	Technicien	20000
20	Ingénieur	36500
20	Technicien	10000
30	Ingénieur	76500
30	Technicien	15600

# Utilisation de la clause HAVING

La clause **WHERE** n'acceptant pas les fonctions de groupe.

La restriction du résultat des fonctions de groupe se fait dans la clause HAVING

## ❖ Syntaxe:

```
SELECT [colonne,] fonction_groupe(argument)
FROM table
[WHERE condition(s) ]
[GROUP BY colonne]
[HAVING condition_condition]
[ORDER BY fonction_group(argument)];
```

## Utilisation de la clause HAVING (2)

### ❖ Remarques:

- La clause **HAVING** peut être utilisée sans présence de fonctions de groupe dans la clause **SELECT**.

### ❖ La différence entre HAVING et WHERE :

- **WHERE** restreint les enregistrements
- **HAVING** restreint les groupes d'enregistrements et peut-être utilisée pour restreindre les enregistrements.

## Utilisation de la clause HAVING (3)

### ❑ Exercice

- Ecrire la requête qui renvoie les numéros de départements (deptno) dont le salaire (sal) maximal (à afficher aussi) est supérieur à 2900.

### ❑ Solution

```
SELECT deptno, MAX(sal)
FROM employe
GROUP BY deptno
HAVING MAX(sal) > 2900;
```

## Utilisation de la clause HAVING (4)

### ❑ Exercice

- Afficher la **somme des salaires** des employés **supérieure** à 5000 par fonction, et les fonctions dont les quatre premières lettres sont différentes de la chaîne de caractères « TECH ». Le résultat est ordonné de façon décroissante sur les

## Utilisation de la clause HAVING (4)

### ❑ Solution

```
SELECT fonction, SUM(sal)
FROM    employe
WHERE    fonction NOT LIKE 'TECH%'
GROUP BY fonction
HAVING   SUM(sal) > 5000
ORDER BY SUM(sal) DESC ;
```



# Les fonctions de groupe imbriquées

## ❑ Exercice

- Ecrire la requête qui affiche la moyenne des salaires sal (de chaque département deptno) la plus élevée.

## ❑ Solution

```
SELECT MAX(AVG(sal))  
FROM employe  
GROUP BY deptno;
```