

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT()

- En SQL, la fonction d'agrégation **COUNT()** permet de compter le nombre d'enregistrements dans une table.
- Connaître le nombre de lignes dans une table est très pratique dans de nombreux cas, par exemple pour savoir combien d'utilisateurs sont présents dans une table

■ Syntaxe

Pour connaître le nombre de lignes totales dans une table, il suffit d'effectuer la requête SQL suivante :

```
SELECT COUNT(*) FROM table;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT()

- Il est aussi possible de connaître le nombre d'enregistrements sur une colonne en particulier.
- Les enregistrements qui possèdent la valeur **NULL** ne seront pas comptabilisés.
- La syntaxe pour compter les enregistrements sur la colonne « **nom_colonne** » est la suivante :

```
SELECT COUNT(nom_colonne) FROM table;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT()

- Il est également possible de compter le nombre d'enregistrements distincts pour une colonne.
- La fonction ne comptabilisera pas les doublons pour une colonne choisie.
- La syntaxe pour compter le nombre de valeurs distinctes pour la colonne « nom_colonne » est la suivante :

```
SELECT COUNT(DISTINCT nom_colonne) FROM table;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT ()

▪ Exemple :

Soit la table « **Client** » décrite comme suit :

id	nom	nbr_commande	ville
1	Allaoui	3	Casa
2	Alami	0	Fes
3	Lotfi	1	Tanger
4	Mandel	7	Casa

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT ()

- **Exemple :**
- Pour compter le nombre total des clients, il suffit d'utiliser COUNT(*) sur toute la table :

```
SELECT COUNT(*) AS nombre_total FROM client;
```

- Le retour de cette requête serait comme suit :

nombre_total
4

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT ()

- **Exemple :**
- Pour compter le nombre de client de Casa, il suffit de faire la même chose mais en filtrant les enregistrements avec WHERE :

```
SELECT COUNT(*) FROM client WHERE ville='Casa';
```

COUNT(*)
2

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: COUNT ()

- **Exemple :**
- L'utilisation de la clause DISTINCT peut permettre de connaître le nombre de villes des clients.
- La requête serait la suivante :

SELECT COUNT(DISTINCT ville) FROM client;

- Le retour de cette requête serait comme suit :

COUNT(DISTINCT ville)
3

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: **AVG ()**

- La fonction d'agrégation **AVG ()** dans le langage SQL permet de calculer la valeur moyenne sur un ensemble d'enregistrements de type numérique et non NULL.
- La syntaxe pour utiliser cette fonction de statistique est simple :

```
SELECT AVG(nom_colonne) FROM nom_table;
```


LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: AVG ()

- Soit la table « **Etudiant** » décrite comme suit :

Id_etud	nom	Moyenne
1	Rahimi	15
2	Alaoui	12
3	Eddamiri	19
4	Fares	16

- Pour calcul de la moyenne de la classe à partir de notre table étudiant, On utilise la requête suivante:

```
SELECT AVG( moyenne ) AS Moyenne_classe FROM etudiant ;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Fonctions d'agrégation: AVG ()

- Le retour de cette requête serait comme suit :

Moyenne_classe
15,5

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Regroupement : GROUP BY

- Soit le MLD suivant :
 - Etudiant(CNE, Nom, Prénom, Ville)
 - Module(Id_Mod, Libellé)
 - Examen(#CNE, #Id_Mod, Note)
- On voudrait répondre à des questions du genre :
 - Quelle est la note moyenne par module ?

SELECT AVG(NOTE) FROM Examen GROUPE BY Id_Mod;

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Regroupement : GROUP BY

- La clause **GROUP BY** est utilisée en SQL pour grouper plusieurs résultats et utiliser une fonction d'agrégation sur un groupe de résultats.
- De façon générale, la commande **GROUP BY** s'utilise de la façon suivante:

```
SELECT colonne1, fonction(colonne2)  
FROM table  
GROUP BY colonne1;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Regroupement : GROUP BY

▪ Exemple

Soit la table « **Achat** » décrite comme suit :

Id_achat	client	tarif	date
1	Pierre	102	2022-01-23
2	Simon	47	2022-01-27
3	Marie	18	2022-02-05
4	Marie	20	2022-02-14
5	Pierre	160	2022-03-03

➤ Pour obtenir le coût total payé par chaque client, il suffit de regrouper selon les clients.

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Regroupement : GROUP BY

▪ Exemple

La requête serait, donc, comme suit :

```
SELECT client, SUM(tarif) FROM Achat  
GROUP BY client;
```

La fonction SUM() permet d'additionner la valeur de chaque tarif pour un même client.

Le résultat sera donc le suivant :

client	SUM(tarif)
Pierre	262
Simon	47
Marie	38

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Regroupement : GROUP BY

▪ Exemple

- Pour connaître le montant moyen payé par chaque client, on peut utiliser une requête qui va utiliser :
 - ✓ GROUP BY pour regrouper les ventes des mêmes clients
 - ✓ La fonction AVG() pour calculer la moyenne des enregistrements
- La requête serait comme suit :

```
SELECT client, AVG(tarif) FROM Achat  
GROUP BY client;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Regroupement : GROUP BY

▪ Exemple

Le retour de cette requête serait comme suit :

client	AVG(tarif)
Pierre	131
Simon	47
Marie	19

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Condition : HAVING

- La condition **HAVING** en SQL est presque similaire à **WHERE** à la seule différence que **HAVING** permet de filtrer en utilisant des fonctions telles que **SUM()**, **COUNT()**, **AVG()**, **MIN()** ou **MAX()**.
- L'utilisation de **HAVING** est comme suit :

```
SELECT colonne1, fonction(colonne2)
FROM nom_table
GROUP BY colonne1
HAVING fonction(colonne2) operateur valeur;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Condition : HAVING

- Soit la table « **Achat** » décrite comme suit :

id	client	tarif	date
1	Pierre	102	2022-01-23
2	Simon	47	2022-01-27
3	Marie	18	2022-02-05
4	Marie	20	2022-02-14
5	Pierre	160	2022-03-03

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Condition : HAVING

- Si on souhaite récupérer la liste des clients qui ont commandé pour un tarif total supérieur à 40, alors il est possible d'utiliser la requête suivante :

```
SELECT client, SUM(tarif)  
FROM Achat  
GROUP BY client  
HAVING SUM(tarif) > 40;
```

client	SUM(tarif)
Pierre	262
Simon	47

SOUS-REQUÊTES (OU REQUÊTES IMBRIQUÉES)

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ **Sous-Requêtes (ou Requêtes Imbriquées)**

- Soit le MLD suivant :
 - Etudiant(CNE, Nom, Prénom, Ville)
 - Module(Id_Mod, Libellé)
 - Examen(#CNE, #Id_Mod, Note)
- On voudrait répondre à des questions du genre :
 - Qui sont les étudiants qui viennent de la même ville que l'étudiant Jalili ?
 - Qui sont les étudiants qui ont une note supérieure à celle de l'étudiant Alaoui ?

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- Dans le langage SQL, une sous-requête (aussi appelée "requête imbriquée" ou "requête en cascade") consiste à exécuter une requête à l'intérieur d'une autre requête.
- Une requête imbriquée est souvent utilisée au sein d'une clause **WHERE** ou de **HAVING** pour remplacer une ou plusieurs constantes.

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

▪ Exemple:

Soit la table Employé décrite comme suit :

id	nom	prenom	DEP_ID	Id_Poste	Salaire
1	Samadi	Said	30	140	7000
2	Fellah	Amina	32	145	8000
3	Alami	Kaouter	42	140	11000
4	Sabbar	Latifa	58	160	5000
5	Farah	Ismail	60	140	15000
6	Raiss	Nabil	80	180	6000

Qui touche un salaire supérieur à **Fellah** ?

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

▪ Exemple:

➤ **Requête principale :**

✓ **Quels employés touchent un salaire supérieur à celui de Fella**h ?

→ **Sous-requête :**

❖ **Quel est le salaire de Fella**h ?

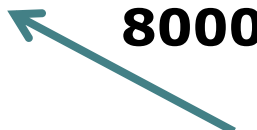
- ✓ La sous-requête (requête interne) s'exécute une fois avant la requête principale.
- ✓ La requête principale (requête externe) utilise le résultat de la sous-requête.

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

▪ Exemple:

```
SELECT nom
FROM Employe
WHERE salaire >=
    (
        SELECT salaire
        FROM Employe
        WHERE nom = 'Fellah'
    );
```



nom
Fellah
Alami
Farah

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

■ Sous-requêtes monolignes

- Renvoient une seule ligne
- Utilisent des opérateurs de comparaison monolignes

Opérateur	Description
=	Égale
<> ou !=	Pas égale
>	Supérieur à
<	Inférieur à
>=	Supérieur ou égale à
<=	Inférieur ou égale à

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

■ Sous-requêtes monolignes

Soit la table Employé décrite comme suit :

id	nom	prenom	DEP_ID	Id_Poste	Salaire
1	Samadi	Said	30	140	7000
2	Fellah	Amina	32	145	8000
3	Alami	Kaouter	42	140	11000
4	Sabbar	Latifa	58	160	5000
5	Farah	Ismail	60	140	15000
6	Raiss	Nabil	80	180	6000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

■ Sous-requêtes monolignes

Pour récupérer le nom, l'Id_Poste et le salaire des employés qui ont le même Id_Poste que l'employé « **Samadi** » et qui ont un salaire supérieur à celui de l'employée « **Fellah** », la requête serait comme suit :

```
SELECT nom, id_Poste, salaire
FROM Employe
WHERE id_Poste =
    (
        SELECT id_Poste
        FROM Employe
        WHERE nom = 'Samadi' )
AND salaire >
    (
        SELECT salaire
        FROM Employe
        WHERE nom = 'Fellah' );
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

■ Sous-requêtes monolignes

Le retour de cette requête serait comme suit :

nom	Id_Poste	Salaire
Alami	140	11000
Farah	140	15000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- **Utiliser des fonctions dans une sous-requête**
- Il est aussi possible d'utiliser les fonctions d'agrégation dans les sous-requêtes.
- **Exemple :**

Pour récupérer le nom, l'Id_Poste et le salaire des personnes qui touchent le salaire minimum, la requête serait :

```
SELECT nom, id_Poste, salaire
FROM Employe
WHERE salaire = (
    SELECT MIN(salaire)
    FROM Employe );
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- Utiliser des fonctions dans une sous-requête

Le retour de cette requête serait :

nom	Id_Poste	Salaire
Sabbar	160	5000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

▪ **HAVING** et sous-requêtes

Pour récupérer le DEP_ID et le salaire des personnes qui touchent le salaire minimum de chaque département mais de telle sorte à ce qu'il soit inférieur ou égal au salaire de l'employé « Raiss », la requête serait :

```
SELECT DEP_ID, MIN(salaire)
FROM Employe
GROUP BY DEP_ID
HAVING MIN(salaire) <= (
    SELECT salaire
    FROM Employe
    WHERE nom = 'Raiss'
);
```


LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

▪ **HAVING** et sous-requêtes

Le retour de cette requête serait comme suit :

DEP_ID	Min(Salaire)
50	6000
58	5000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

■ Sous-requêtes multilignes

Ce sont des requêtes qui retournent plusieurs lignes et qui utilisent des opérateurs de comparaison multilignes.

Opérateur	Description
IN	Égale à une des valeurs dans la liste
ALL	Permet de vérifier si une valeur est supérieure, inférieure, ou égale à toutes les valeurs renvoyées par la sous-requête
ANY	Permet de vérifier si une valeur est supérieure, inférieure, ou égale à au moins une des valeurs retournées par la sous-requête

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

▪ **ANY** dans les sous-requêtes multilignes

Soit la table Employé décrite comme suit :

id	nom	prenom	DEP_ID	Id_Poste	Salaire
1	Samadi	Said	50	140	7000
2	Fellah	Amina	50	145	8000
3	Alami	Kaouter	42	140	11000
4	Sabbar	Latifa	58	160	5000
5	Farah	Ismail	42	140	15000
6	Raiss	Nabil	50	180	6000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- **ANY** dans les sous-requêtes multilignes

Pour récupérer l'Id, le nom, l'Id Poste et le salaire des employés qui ont un Id_Poste différent de 140 et un salaire inférieur au salaire d'au moins un employé ayant l'Id_Poste 140, la requête serait :

```
SELECT id, nom, Id_Poste, salaire
FROM Employe
WHERE id_Poste != 140
AND salaire < ANY (
    SELECT salaire
    FROM Employe
    WHERE id_Poste = 140) ;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- **ANY** dans les sous-requêtes multilignes

Le retour de cette requête serait :

id	nom	Id_Poste	Salaire
2	Fellah	145	8000
4	Sabbar	160	5000
6	Raiss	180	6000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- **ALL** dans les sous-requêtes multilignes

Pour récupérer l'Id, le nom, l'Id_Poste et le salaire des employés qui ont un Id_Poste différent de 140 et un salaire inférieur au salaire de tous les employés ayant l'Id_Poste 140, la requête serait :

```
SELECT id, nom, id_Poste, salaire  
FROM Employe  
WHERE id_Poste != 140  
AND salaire < ALL (
```

```
SELECT salaire  
FROM Employe  
WHERE id_Poste = 140 ) ;
```

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- **ALL** dans les sous-requêtes multilignes

Le retour de cette requête serait :

id	nom	Id_Poste	Salaire
4	Sabbar	160	5000
6	Raiss	180	6000

LANGAGE D'INTERROGATION DE DONNÉES SQL (LID)

⌘ Sous-Requêtes (ou Requêtes Imbriquées)

- **IN** dans les sous-requêtes multilignes

Pour récupérer l'Id, le nom, l'Id Poste et le salaire des employés qui ont un Id Poste différent de 140 et un salaire égal à un des salaires des employés ayant l'Id Poste 140, la requête serait :

```
SELECT id, nom, id_Poste, salaire
FROM Employe
WHERE id_Poste != 140
AND salaire IN (
    SELECT salaire
    FROM Employe
    WHERE id_Poste = 140 ) ;
```