

# Consultation de données

Introductions aux Bases de Données  
Nathanaël Martel

# *Consultation des données*

- **Pour récupérer des données dans la base, il faut utiliser une requête commençant par le mot clés « SELECT », que ce soit :**
  - Dans un ordre ou un autre,
  - Tout le résultat ou seulement les enregistrements dont la valeur répond à un critère,
  - Certains champs et pas les autres,
  - Sur une table ou sur plusieurs table,
  - En faisant des transformations, des calculs ou des regroupements...
- **Le SGBD renvoie une collection de données**

# Consultation simple

- Pour récupérer toute une table

```
SELECT * FROM `auteur`;
```

id	nom	prenom	date_naissance	date_mort
1	Jean	Cocteau	NULL	NULL
2	Hugo	Victor	NULL	NULL
3	Voltaire		NULL	NULL
4	De Musset	Alfred	NULL	NULL
5	De Balzac	Honoré	NULL	NULL
6	Shakespeare	William	NULL	NULL
7	Balzac	Honoré	NULL	NULL
8	Alain	Emile Chartier	NULL	NULL
9	Valéry	Paul	NULL	NULL
10	Bernard	Tristan	NULL	NULL

# Consultation simple

- Pour récupérer les n premiers enregistrements :
- LIMIT n

```
SELECT * FROM `livre` LIMIT 5;
```

id	titre	auteur_id	date_achat
102	Opium	1	NULL
103	Essai De Critique Indirecte	1	NULL
104	Secrets De Beauté	1	NULL
105	Orphée	1	NULL
106	Les Parents Terribles	1	NULL

5 rows in set (0,00 sec)

# Consultation avec condition

- Pour récupérer le ou les enregistrements correspondant à une valeur de champs
- WHERE **champ**='valeur'

```
SELECT * FROM `livre` WHERE id=110;
```

```
SELECT * FROM `livre` WHERE auteur_id=2;
```

```
SELECT * FROM `livre` WHERE titre='Cromwell';
```

```
SELECT * FROM `livre`  
WHERE titre='Cromwell' AND auteur_id=2;
```

# *Consultation avec condition*

- Différent : **champ** **!=** 'valeur'
- Supérieur strict : **champ** **>** valeur
- Supérieur ou égale : **champ** **>=** valeur
- Inférieur strict : **champ** **<** valeur
- Inférieur ou égale : **champ** **<=** valeur

# Consultation avec condition

- **champ** LIKE '**va**leur'
- «%» remplace un nombre indéterminé de caractères
- «\_» remplace exactement un caractère

```
SELECT * FROM `livre` WHERE titre LIKE '%homme%';
```

# *Consultation avec condition*

- **champ** IN ('valeur1', 'valeur2')

```
SELECT * FROM `livre` WHERE auteur_id IN (7, 5);
```



# *Consultation avec condition*

- **champ** IS NULL / **champ** IS NOT NULL

```
SELECT * FROM `auteur` WHERE date_mort IS NULL;
```

# *Consultation dans un certain ordre*

- Pour récupérer les enregistrements triés dans un ordre donné
- ORDER BY **champ**
- ORDER BY **champ** DESC
- ORDER BY **champ1** ASC **champ2** ASC

```
SELECT * FROM `livre` ORDER BY titre;
```

```
SELECT * FROM `auteur`  
ORDER BY `nom` ASC, `prenom` DESC;
```

# *Consultation de certains champs*

- Pour récupérer une partie des champs

```
SELECT titre FROM `livre`;
```

```
SELECT prenom, nom FROM `auteur`;
```

C'est utile dans deux cas :

- Quand le serveur est distant et que les données sont lourdes
- Quand il y a plusieurs tables et que les champs peuvent être ambiguë

# Consultation dans plusieurs table

- On parle de : Jointure
- Elle peut se faire par la droite ou par la gauche en incluant ou non les enregistrement qui ne sont que d'un côté de la jointure
- On peut utiliser un Alias, qui n'existera que le temps de la requête

```
SELECT * FROM livre, auteur WHERE livre.auteur_id=auteur.id;
```

```
SELECT * FROM livre  
LEFT OUTER JOIN auteur ON (livre.auteur_id=auteur.id);
```

```
SELECT * FROM livre AS l  
LEFT OUTER JOIN auteur AS a ON (l.auteur_id=a.id);
```

```
SELECT l.titre, a.prenom, a.nom FROM livre AS l  
LEFT OUTER JOIN auteur AS a ON (l.auteur_id=a.id)  
WHERE a.id IN (5,7)  
ORDER BY l.titre;
```

# Transformation et Calcul

```
SELECT 1+1;
```

```
SELECT count(*) AS nb_livre FROM livre;
```

```
→ count(*), sum(champs), min(), max(), avg()
```

```
SELECT NOW();
```

```
SELECT * FROM livre  
WHERE DATEDIF(date_achat, NOW()) < 90;
```

# Regroupement

- Récupérer uniquement les enregistrements dont un champs est différents

```
SELECT auteur_id, count(*) AS nb_livre FROM livre  
GROUP BY auteur_id;
```

```
SELECT CONCAT(a.prenom, ' ', a.nom) AS nom, count(*) AS  
nb_livre  
FROM livre AS l LEFT OUTER JOIN auteur AS a ON  
(l.auteur_id=a.id)  
GROUP BY auteur_id;
```

# *Union de collections*

- Il est possible de ré-unir deux selections dans un même résultat.

```
(SELECT nom, prenom, 'etudiant' AS statut  
FROM etudiant)  
UNION  
(SELECT nom, prenom, 'enseignant' AS statut  
FROM enseignant)
```

# Résumé

```
SELECT select_expr [, select_expr ...]  
  [FROM table_references  
    [LEFT OUTER JOIN partition_list]  
  [WHERE where_condition]  
  [GROUP BY {col_name | expr} ]  
  [ORDER BY {col_name | expr}  
    [ASC | DESC], ...]  
  [LIMIT {[offset,] row_count]
```



# Conclusion

SELECT permet de récupérer des données dans la base.

```
SELECT * FROM table_name  
WHERE condition ORDER BY col_name ASC|DESC  
GROUP BY col_name LIMIT offset, row_count
```