Base de données
Troisième partie

Aide mysql

- Dans l'aide de mysql on trouve :

Aide mysql

- Dans l'aide de mysql on trouve :
- ```
SELECT [STRAIGHT_JOIN]
       [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
       [SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS] [HIGH_PRIORITY]
       [DISTINCT | DISTINCTROW | ALL]
    select_expression,...
    [INTO {OUTFILE | DUMPFILE} 'nom_fichier' export_options]
    [FROM table_references
      [WHERE where_definition]
      [GROUP BY {unsigned_integer | nom_de_colonne | formula}
                                                [ASC | DESC], ...
      [HAVING where_definition]
      [ORDER BY {unsigned_integer | nom_de_colonne | formula}
                                                [ASC | DESC] ,...]
      [LIMIT [offset,] lignes]
      [PROCEDURE procedure_name(argument_list)]
      [FOR UPDATE | LOCK IN SHARE MODE]]
```
```

Aide mysql

- Il y a beaucoup de possibilités que nous n'utiliserons pas.

Aide mysql

- Il y a beaucoup de possibilités que nous n'utiliserons pas.
- On va se limiter à

Aide mysql

- Il y a beaucoup de possibilités que nous n'utiliserons pas.
- On va se limiter à
  - ```
    SELECT [DISTINCT]
        <select_expression,...>
        FROM <table_references>
          [WHERE <condition> ]
          [GROUP BY <att> [ASC | DESC], ...]
          [HAVING <condition>]
          [ORDER BY <att> [ASC | DESC] ,...]
          [LIMIT <n> [OFFSET <p>]]
    ```
    ```

Aide mysql

- Dans le `<select_expression,...>`

Aide mysql

- Dans le `<select_expression,...>`
- on place les noms des attributs (projection)

Aide mysql

- Dans le `<select_expression,...>`
- on place les noms des attributs (projection)
- éventuellement renommés (`AS`)

Aide mysql

- Dans le `<select_expression,...>`
- on place les noms des attributs (projection)
- éventuellement renommés (`AS`)
- et/ou les fonctions agrégatives (`COUNT, MIN, MAX, ...`)

Aide mysql

- Dans le `<table_references>`

Aide mysql

- Dans le `<table_references>`
- on place les noms des tables

Aide mysql

- Dans le `<table_references>`
- on place les noms des tables
- éventuellement renommées (`AS`)

Aide mysql

- Dans le `<table_references>`
- on place les noms des tables
- éventuellement renommées (`AS`)
- éventuellement jointées.

Principe

- Une fonction agrégative s'applique à l'ensemble des valeurs d'un attribut `att` des enregistrements renvoyés par une requête ;

Principe

- Une fonction agrégative s'applique à l'ensemble des valeurs d'un attribut `att` des enregistrements renvoyés par une requête ;
- Elle agrège donc ces enregistrements en un seul paquet, pour en calculer :

Principe

- Une fonction agrégative s'applique à l'ensemble des valeurs d'un attribut `att` des enregistrements renvoyés par une requête ;
- Elle agrège donc ces enregistrements en un seul paquet, pour en calculer :
- le cardinal (`COUNT(att)`) et `COUNT(*)`

Principe

- Une fonction agrégative s'applique à l'ensemble des valeurs d'un attribut `att` des enregistrements renvoyés par une requête ;
- Elle agrège donc ces enregistrements en un seul paquet, pour en calculer :
- le cardinal (`COUNT(att)`) et `COUNT(*)`
- la somme (`SUM(att)`) ;

Principe

- Une fonction agrégative s'applique à l'ensemble des valeurs d'un attribut `att` des enregistrements renvoyés par une requête ;
- Elle agrège donc ces enregistrements en un seul paquet, pour en calculer :
- le cardinal (`COUNT(att)`) et `COUNT(*)`
- la somme (`SUM(att)`) ;
- la valeur minimale/maximale (`MIN(att)`, `MAX(att)`)

Principe

- Une fonction agrégative s'applique à l'ensemble des valeurs d'un attribut `att` des enregistrements renvoyés par une requête ;
- Elle agrège donc ces enregistrements en un seul paquet, pour en calculer :
- le cardinal (`COUNT(att)`) et `COUNT(*)`
- la somme (`SUM(att)`) ;
- la valeur minimale/maximale (`MIN(att)`, `MAX(att)`)
- la moyenne (`AVG(att)`), la variance (`VARIANCE(att)`)

Sous-paquets : `GROUP BY`

- On peut demander dans la requête à regrouper dans des sous-paquets les enregistrements ayant la même valeur pour un attribut

Sous-paquets : `GROUP BY`

- On peut demander dans la requête à regrouper dans des sous-paquets les enregistrements ayant la même valeur pour un attribut
- Cela permet d'appliquer la fonction agrégative sur chaque sous-paquet, plutôt que sur l'ensemble des enregistrements

Sous-paquets : `GROUP BY`

- On peut demander dans la requête à regrouper dans des sous-paquets les enregistrements ayant la même valeur pour un attribut
- Cela permet d'appliquer la fonction agrégative sur chaque sous-paquet, plutôt que sur l'ensemble des enregistrements
- En général on affiche également l'attribut qui a servi au regroupement

Sous-paquets : GROUP BY

- ```
  SELECT Continent, SUM(Population)
  FROM Country GROUP BY Continent;
  +---------------+------------------+
  | Continent     | SUM(Population)  |
  +---------------+------------------+
  | Asia          |       3705025700 |
  | Europe        |        730074600 |
  | North America |        482993000 |
  | Africa        |        784475000 |
  | Oceania       |         30401150 |
  | Antarctica    |                0 |
  | South America |        345780000 |
  +---------------+------------------+
  7 rows in set (0.00 sec)
  ```
  ```

Sous-paquets : GROUP BY

- Attention au comportement si on met un attribut qui n'a pas la même valeur pour tous les enregistrements d'un même paquet (ce qui n'a d'ailleurs aucun sens...) :

```
SELECT Continent, Name, SUM(Population)
FROM Country GROUP BY Continent;
```

Sous-paquets : GROUP BY

```
SELECT Continent, Name, SUM(Population)
FROM Country GROUP BY Continent;


+---------------+----------------+-----------------+
| Continent     | Name           | SUM(Population) |
+---------------+----------------+-----------------+
| Asia          | Afghanistan    |      3705025700 |
| Europe        | Albania        |       730074600 |
| North America | Aruba          |       482993000 |
| Africa        | Angola         |       784475000 |
| Oceania       | American Samoa |        30401150 |
| Antarctica    | Antarctica     |               0 |
| South America | Argentina      |       345780000 |
+---------------+----------------+-----------------+
7 rows in set (0.00 sec)
```

mysql renvoie la valeur du premier enregistrement du sous-paquet !

HAVING

- Le résultat d'une fonction d'agrégation ne peut pas servir dans une condition de sélection (`WHERE`). Pour filtrer suivant des agrégats on utilise `HAVING`.

HAVING

- Le résultat d'une fonction d'agrégation ne peut pas servir dans une condition de sélection (`WHERE`). Pour filtrer suivant des agrégats on utilise `HAVING`.
- Par exemple : pour ne tenir compte que des continents dont la population n'est pas nulle :

HAVING

- Le résultat d'une fonction d'agrégation ne peut pas servir dans une condition de sélection (`WHERE`). Pour filtrer suivant des agrégats on utilise `HAVING`.
- Par exemple : pour ne tenir compte que des continents dont la population n'est pas nulle :

```
SELECT Continent, SUM(Population)
FROM Country
GROUP BY Continent
HAVING SUM(Population) > 0;
```

HAVING

- Le résultat d'une fonction d'agrégation ne peut pas servir dans une condition de sélection (`WHERE`). Pour filtrer suivant des agrégats on utilise `HAVING`.
- Par exemple : pour ne tenir compte que des continents dont la population n'est pas nulle :
  - ```
    SELECT Continent, SUM(Population)
    FROM Country
    GROUP BY Continent
    HAVING SUM(Population) > 0;
    ```
-
    ```

HAVING

```
SELECT Continent, SUM(Population)
FROM Country
GROUP BY Continent
HAVING SUM(Population) > 0;
```

Continent	Name	SUM(Population)
Asia	Afghanistan	3705025700
Europe	Albania	730074600
North America	Aruba	482993000
Africa	Angola	784475000
Oceania	American Samoa	30401150
South America	Argentina	345780000

Principe

- D'un point de vue logiciel, un SGBD doit réaliser trois grandes tâches :

Principe

- D'un point de vue logiciel, un SGBD doit réaliser trois grandes tâches :
- Stockage et accès aux données

Principe

- D'un point de vue logiciel, un SGBD doit réaliser trois grandes tâches :
- Stockage et accès aux données
- Logique applicative (propre à chaque entreprise utilisant un SGBD). C'est le reflet de l'organisation informationnelle de l'entreprise

Principe

- D'un point de vue logiciel, un SGBD doit réaliser trois grandes tâches :
- Stockage et accès aux données
- Logique applicative (propre à chaque entreprise utilisant un SGBD). C'est le reflet de l'organisation informationnelle de l'entreprise
- Présentation

Principe

- Par nature un SGBD doit centraliser les données sans les dupliquer (intégrité des données)

Principe

- Par nature un SGBD doit centraliser les données sans les dupliquer (intégrité des données)
- En revanche les données doivent être accessibles par des utilisateurs localisés en des endroits différents

Principe

- Par nature un SGBD doit centraliser les données sans les dupliquer (intégrité des données)
- En revanche les données doivent être accessibles par des utilisateurs localisés en des endroits différents
- Solution : architecture client/serveur

Principe

- Par nature un SGBD doit centraliser les données sans les dupliquer (intégrité des données)
- En revanche les données doivent être accessibles par des utilisateurs localisés en des endroits différents
- Solution : architecture client/serveur
- Le serveur est chargé du stockage et de l'accès aux données, et éventuellement d'une petite part de la logique applicative

Principe

- Par nature un SGBD doit centraliser les données sans les dupliquer (intégrité des données)
- En revanche les données doivent être accessibles par des utilisateurs localisés en des endroits différents
- Solution : architecture client/serveur
- Le serveur est chargé du stockage et de l'accès aux données, et éventuellement d'une petite part de la logique applicative
- Le client, au plus près de l'utilisateur, assure le plus gros de la logique applicative et la présentation (interface avec l'utilisateur)

Inconvénient

- Le principal inconvénient de cette structure est le coût de la maintenance et de l'évolution du système

Inconvénient

- Le principal inconvénient de cette structure est le coût de la maintenance et de l'évolution du système
- Si la logique applicative est modifiée, il faut modifier tous les clients (très couteux)

Solution

- Une solution est l'architecture 3-tiers qui comporte

Solution

- Une solution est l'architecture 3-tiers qui comporte
- un serveur, assurant toujours le stockage et l'accès aux données

Solution

- Une solution est l'architecture 3-tiers qui comporte
- un serveur, assurant toujours le stockage et l'accès aux données
- un intermédiaire : le middleware, chargé de la logique applicative

Solution

- Une solution est l'architecture 3-tiers qui comporte
- un serveur, assurant toujours le stockage et l'accès aux données
- un intermédiaire : le middleware, chargé de la logique applicative
- des clients, chargés essentiellement de la présentation, souvent sous la forme de navigateur WEB

Solution

- Une solution est l'architecture 3-tiers qui comporte
- un serveur, assurant toujours le stockage et l'accès aux données
- un intermédiaire : le middleware, chargé de la logique applicative
- des clients, chargés essentiellement de la présentation, souvent sous la forme de navigateur WEB
- Des études de cabinets d'audit montrent que l'architecture 3-tiers est plus avantageuse pour la maintenance et l'évolution, dès que les systèmes d'informations sont un peu conséquents.