

Cours Web

MySQL

Lionel Seinturier

Université Pierre & Marie Curie

Lionel.Seinturier@lip6.fr



11/7/02

Web

253

Lionel Seinturier

11. MySQL

Structured Query Language

Langage de manipulation des données stockées dans une base de données

- interrogation/insertion/modification/suppression
- gestion des droits d'accès

Principe

- organisation des données en **base** = espace de stockage
- principale structure stockée dans une base
table = ensemble de données ayant un lien logique entre elles

Exemple : base "gestion du personnel" contenant tables "employés", "congés"

Les données d'une table **réfèrent** éventuellement les données d'autres tables
⇒ SGBDR : système de gestion de bases de données relationnel

Exemple : Oracle, SQL Server, Access, PostGres, MySQL, ...

Web

254

Lionel Seinturier

11. MySQL

MySQL

Fonctionne selon le mode client/serveur



MySQL
client

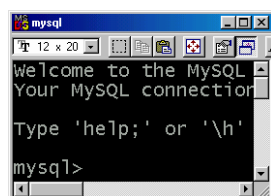
requête SQL

protocole MySQL

réponse



MySQL
serveur



Web

255

Lionel Seinturier

11. MySQL

Création/suppression de bases

```
CREATE DATABASE nombase ;  
DROP DATABASE nombase ;
```

Tables

- chaque table est définie par un ensemble d'**attributs**
- chaque attribut à un **type** (chaîne, entier, réel, booléen, date, ...)
- un enregistrement dans une table (= une ligne) est appelé un **tuple**

Attributs

Nom	Adresse	Salaire	Age

n-uplets
ou tuples

Web

256

Lionel Seinturier

11. MySQL

Interrogation/extraction de données dans des tables

Principe des commandes

```
SELECT  $attribut_1, \dots, attribut_n$ 
FROM  $table_1, \dots, table_p$ 
WHERE  $prédicat$  ;
```

- SELECT : un ensemble d'attributs \in aux tables
- FROM : un ensemble de tables de la base
- WHERE : un prédicat pour sélectionner les tuples

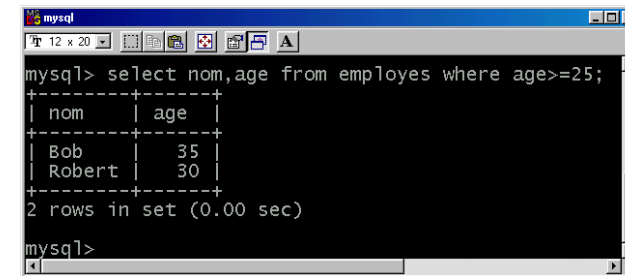
Prédicat construit à partir

- des attributs mentionnés dans la clause SELECT
- d'opérateurs de comparaison (< > >= <= = <>)
- d'opérateurs booléens (AND OR NOT)

- ⇒ tous les tuples qui vérifient le prédicat sont affichés à l'écran
- ⇒ le résultat peut être vide

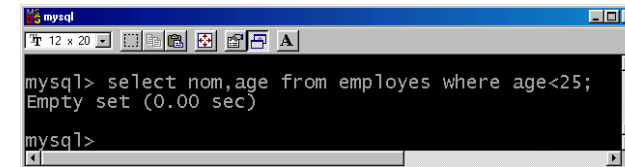
11. MySQL

Interrogation/extraction de données dans des tables



```
mysql> select nom,age from employes where age>=25;
+----+-----+
| nom | age |
+----+-----+
| Bob | 35  |
| Robert | 30 |
+----+-----+
2 rows in set (0.00 sec)

mysql>
```



```
mysql> select nom,age from employes where age<25;
Empty set (0.00 sec)

mysql>
```

11. MySQL

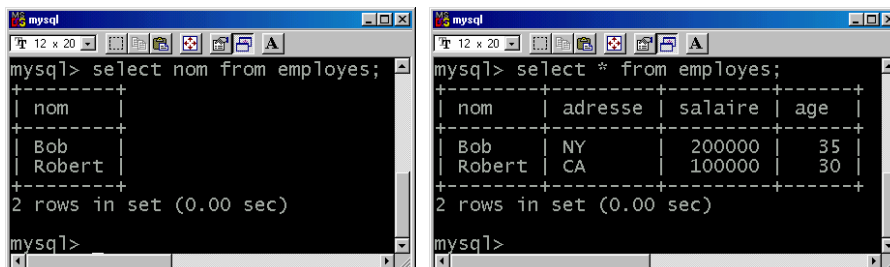
Interrogation/extraction de données dans des tables

Clause WHERE facultative

- ⇒ sélection de tous les tuples de la table

La liste d'attributs du SELECT peut être remplacée par *

- ⇒ sélection de tous les attributs de la table



```
mysql> select nom from employes;
+----+
| nom |
+----+
| Bob |
| Robert |
+----+
2 rows in set (0.00 sec)

mysql>

mysql> select * from employes;
+----+-----+-----+-----+
| nom | adresse | salaire | age |
+----+-----+-----+-----+
| Bob | NY      | 200000  | 35  |
| Robert | CA      | 100000  | 30  |
+----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

11. MySQL

Création/suppression de tables

CREATE TABLE $nomtable$ ($attribut_1 type_1, \dots, attribut_n type_n$) ;

ex : CREATE TABLE employes

(nom varchar(20), adresse varchar(30), salaire float, age int) ;

Types de données

- int, float, varchar(n), text (n max. 255, text max 65535)
- numeric(m,d), datetime, date, time, blob, enum(v_1, \dots, v_n)
- tinyint, smallint, mediumint, bigint, double

Suppression de table

DROP TABLE $nomtable$;

11. MySQL

Création/suppression de tables

Définition de clés primaires

- attributs \in clé NOT NULL
- clause PRIMARY KEY

ex : CREATE TABLE adresses
(nom varchar(20) **NOT NULL**, nom varchar(20) **NOT NULL**,
adresse varchar(30), **PRIMARY KEY(nom,prenom)**);

Champs auto-incrémenté

ex : CREATE TABLE cd
(id int AUTO_INCREMENT, titre varchar(20), interprete varchar(20),
PRIMARY KEY(id));

11. MySQL

Ajout/modification de tuple dans une table

INSERT INTO *nomtable* VALUES (val_1 , ..., val_n);

ex : INSERT INTO employes VALUES ('John', 'MD', 150000.00, 32);

- autant de valeurs que d'attributs (sinon erreur)
- chaînes entre apostrophes

Modification de tuples

UPDATE *nomtable* SET $attribut_1 = val_1$, ..., $attribut_n = val_n$
WHERE *predicat* ;

ex : UPDATE employes SET salaire = salaire + 1000
WHERE adresse = 'CA' ;

- éventuellement +sieurs tuples modifiés par la commande

11. MySQL

Suppression de tuple dans une table

DELETE FROM *nomtable* WHERE *predicat* ;

ex : DELETE FROM employes WHERE adresse = 'CA' ;

- éventuellement +sieurs tuples supprimés par la commande

11. MySQL

Compléments sur la clause SELECT

Tri du résultat

SELECT $attribut_1$, ..., $attribut_n$ FROM ... WHERE ...
ORDER BY $attribut_1$, ..., $attribut_p$;

- affichage des tuples selon l'ordre spécifié par la clause ORDER BY
- par défaut ordre ascendant (alphabétique ou numérique)
- mot clé DESC après l'attribut = ordre descendant
- les attributs de la clause ORDER BY ne sont pas forcément ceux du SELECT

11. MySQL

Compléments sur la clause SELECT

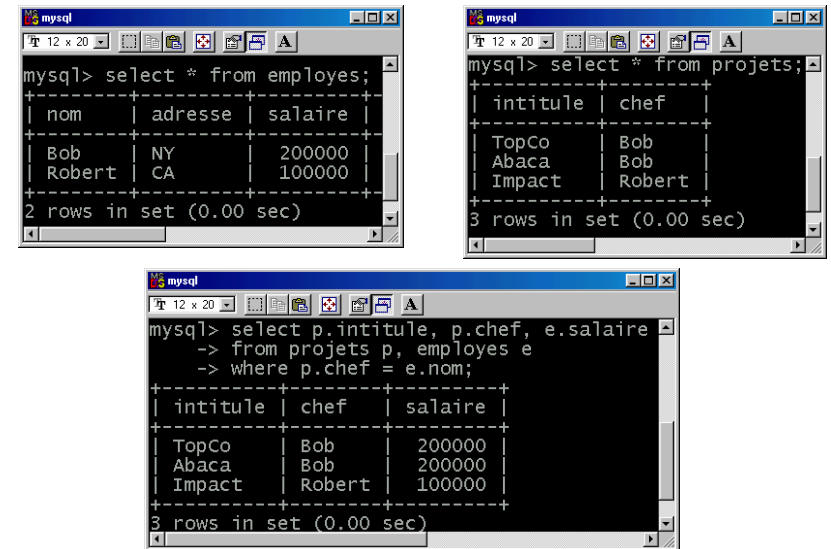
Requêtes multi relations

- extraction de données parmi +sieurs tables
- mise en relation (jointure) entre +sieurs attributs pris chacun dans une table
- ces attributs ne sont pas forcément ceux du SELECT
- utilisation d'alias pour différencier les ≠ tables
- les alias peuvent être omis en cas de non ambiguïté

```
SELECT alias1.attribut1, ... , aliasp.attributn
FROM   table1 alias1, ... , tablep aliasp
WHERE  predicat ;
```

ex : SELECT p.intitule, p.chef, e.salaire
 FROM projets p, employes e
 WHERE p.chef = e.nom ;

11. MySQL



11. MySQL

Compléments sur la clause SELECT

Requêtes imbriquées

- utilisation d'une requête SELECT imbriquée dans la clause WHERE
- +sieurs niveaux d'imbriication possible
- utilisation d'opérateurs : IN, EXISTS ou NOT EXISTS

Opérateur IN

```
SELECT ... FROM ...
WHERE  attribut1, ... , attributn IN
      ( SELECT ... FROM ... );
```

- ⇒ les n attributs doivent ∈ aux tuples sélectionnés par le SELECT imbriqué
- ⇒ autant d'attributs dans le SELECT imbriqué que dans le WHERE
- ⇒ les types des attr. du SELECT imbriqué doivent correspondre à ceux des attr. du WHERE

11. MySQL

Compléments sur la clause SELECT

Requêtes imbriquées IN

employes (nom, adresse, salaire, age)
projets (intitule, chef, budget, adresse)

Quels sont les chefs de projet qui habitent dans des états
dans lesquels il y a des projets dont le budget est > à 1 M\$?

```
SELECT chef
FROM   projets
WHERE  adresse IN
      ( SELECT adresse
        FROM projets
        WHERE budget > 1000000 );
```

11. MySQL

Compléments sur la clause SELECT

Requêtes imbriquées EXISTS / NOT EXISTS

```
SELECT ... FROM ...  
WHERE EXISTS  
    ( SELECT * FROM ... );
```

- EXISTS est un quantificateur existentiel
- le WHERE est vérifié si le SELECT imbriqué sélectionne **au moins un** tuple

```
SELECT ... FROM ...  
WHERE NOT EXISTS  
    ( SELECT * FROM ... );
```

- le WHERE est vérifié si le SELECT imbriqué sélectionne **aucun** tuple

11. MySQL

Compléments sur la clause SELECT

Requêtes imbriquées EXISTS

employes (nom, adresse, salaire, age)
projets (intitule, chef, budget, adresse)

Nom des employés qui habitent dans un état où il y a un projet ?

```
SELECT nom  
FROM   employes e  
WHERE EXISTS  
    ( SELECT *  
      FROM projets p  
      WHERE p.adresse = e.adresse );
```

11. MySQL

Compléments sur la clause SELECT

Fonctions d'aggrégation

- permettent d'effectuer certains calculs à partir d'une sélection
- fonctions disponibles : SUM, MAX, MIN, AVG (moyenne), COUNT (nombre)

```
SELECT fonction(attribut1) , ... , fonction(attributn)  
FROM   table1 , ... , tablep  
WHERE  prédicat ;
```

employes (nom, adresse, salaire, age)
projets (intitule, chef, budget, adresse)

Budget moyen des projets en Californie ?

```
SELECT AVG(budget)  
FROM   projets  
WHERE  adresse = 'CA' ;
```

11. MySQL

Compléments sur la clause SELECT

Groupements

- permettent de grouper les tuples sélectionnés en fonction d'attributs

```
SELECT attribut1 , ... , attributn  
FROM   table1 , ... , tablep  
WHERE  prédicat  
GROUP BY attribut1 , ... , attributp
```

- sélectionne les *n* attr. en les groupant
selon l'ordre spécifié par les *p* attr. du GROUP BY

ex : liste des chefs de projet groupée par état