INFO-H-403 Bases de données
Séance d'exercices 7
SQL (1)

7 novembre 2011

SQL

- Structured Query Language
- Langage de définition de de manipulation de données relationnelles

SQL DDL (Data Definition Language)

- Sous ensemble de SQL pour définir la structure de la base de données.

Création d'une table

```
CREATE TABLE Employee (
  SSN        VARCHAR(9)        NOT NULL,
  Name       VARCHAR(15)       NOT NULL,
  SuperSSN   VARCHAR(9),
  PRIMARY KEY (SSN),
  FOREIGN KEY (SuperSSN) REFERENCES Employee(SSN)
)
```

Suppression d'une table

```
DROP TABLE Employee
```

SQL DDL

Ajout d'une colonne

```
ALTER TABLE Employee
      ADD BDate DATE
```

Suppression d'une colonne

```
ALTER TABLE Employee
      DROP BDate
```

Cette requête ne fonctionne pas avec SQLite.

Requêtes SQL

Structure générale

```
SELECT attributs
FROM relations
WHERE conditions
```

- Les attributs et les relations sont séparés par des virgules.
- Dans la clause `WHERE`, on peut placer :
  - des comparaisons binaires d'attributs et de valeurs
  - des connecteurs `AND`, `OR`, `NOT`
  - attribut `IN` ensemble de valeurs
  - `EXISTS(` sous-requête `)`

Requêtes SQL

Adresse et date de naissance des employés nommés John Smith

```
SELECT e.Address, e.BDate
FROM Employee e
WHERE e.FName='John' AND e.LName='Smith'
```

Jointure : Noms des employés du département recherche

```
SELECT e.FName
FROM Employee e, Department d
WHERE e.DNum = d.DNum AND d.DName='Research'
```

Remarquez l'utilisation de variables (alias).

Requêtes SQL : inclusion

Noms des employés du département 1, 2 ou 3

```
SELECT e.FName
FROM Employee e
WHERE e.Dno IN {1,2,3}
```

Noms des employés des départements dirigés par Bill

```
SELECT e.FName
FROM Employee e
WHERE e.Dno IN (SELECT d.Dno
                FROM Department d
                WHERE d.Manager = 'Bill')
```

Requêtes SQL : existence

Noms des employés qui ont au moins un dépendant

```
SELECT e.FName
FROM Employee e
WHERE EXISTS  ( SELECT *
                FROM Dependent d
                WHERE e.SSN = d.ESSN )
```

Requêtes SQL : divers

- 'SELECT *' renvoie toutes les colonnes
- 'SELECT DISTINCT' renvoie les tuples distincts
- Le mot clé 'UNION' fait l'union de deux tables ayant le même nombre de colonnes en supprimant les doublons.
- La clause 'ORDER BY attributs' trie les résultats

Quantificateur universel

Formes équivalentes :

- Les employés qui travaillent sur tous les projets.
- Les employés pour lesquels il n'existe pas de projet sur lequel ils ne travaillent pas.

Traduction en SQL

```
SELECT e.FName, e.LName
FROM Employee e
WHERE NOT EXISTS
  ( SELECT *
    FROM Project p
    WHERE NOT EXISTS
      ( SELECT *
        FROM WorksOn w
        WHERE w.ESSN = e.SSN AND w.PNo = p.PNumber) )
```

Requêtes SQL : modifications

Insertion d'une ligne

```
INSERT
INTO Employee(SSN, Name)
VALUES (9857234, 'John')
```

Modification d'une ligne

```
UPDATE Employee
SET Name = 'Bill'
WHERE SSN = 9857234
```

Suppression d'une ligne

```
DELETE FROM Employee
WHERE SSN = 9857234
```

SQLite en labo (ou chez vous)

- Lancer *SQLite Database Browser*
- Ouvrir la base de données `puf.sqlite` trouvée sur `cs.ulb.ac.be/public/teaching/infoh303/tp`
- Ecrire les requêtes et vérifier les résultats

Attention, pour utiliser les contraintes de clé étrangère, à chaque connexion, il faut introduire le pragma

```
PRAGMA foreign_keys = ON;
```