ESIR1 BD
Bases de données

requêtes complexes

Olivier Ridoux

Plan de la séance

- Requêtes imbriquées
- Simulation de la division relationnelle

Requêtes imbriquées

Motivation

- SELECT ... FROM ... WHERE ...

$$\Pi_{\ldots}(\sigma_{\ldots}(\ldots))$$

...mais l'AR permet toutes les imbrications...

...et SQL aussi !

Nomenclature des tables (1)

- Table : n lignes × m colonnes

  SELECT ... FROM ... WHERE ...

- Ligne : 1 ligne × m colonnes

  SELECT ... FROM ... WHERE key = ...

  SELECT agrégat ... FROM ... WHERE ...

- Colonne : n lignes × 1 colonne

  SELECT colonne FROM ... WHERE ...

Nomenclature des tables (2)

- Valeur : 1 ligne × 1 colonne

SELECT colonne WHERE key = ...

SELECT agregat ...

- Booléen : 0 ligne × 0 colonne

SELECT colonne ... WHERE faux

Imbrication de requête (1)

SELECT ...

FROM (requête table) t

...

- Obligation de renommer la table

Exemple (1)

```
SELECT Nom, Prenom, Intitule
FROM Adherent adh,
        ( SELECT NumAdherent, NumActivite
          FROM AdherentActivite adac
          WHERE adac.Expertise > 5) adax,
        Activite act
WHERE adh.NumAdherent = adax.NumAdherent
   AND adac.NumActivite = act.NumActivite
```

Imbrication de requête (2)

SELECT …

FROM …

WHERE attr IN (requête colonne)

- Aussi ALL, SOME (= ANY)

…WHERE arg relop ALL (requête colonne)

…WHERE arg relop ANY (requête colonne)

Exemple (2)

```
SELECT Intitule
FROM Activite
WHERE NumActivite
      IN ( SELECT NumActivite
           FROM AdherentActivite
           WHERE expertise > 5 ) ;
```

Imbrication de requête (2,5)

- IN ≡ = ANY
- NOT IN ≡ <> ALL

Imbrication de requête (3)

SELECT ... FROM ...
WHERE attr relop (requête valeur)

SELECT f(... requête valeur) FROM ...

Imbrication simple (1)

- La requête interne **ne fait pas référence** à des éléments de la requête externe...

...elle en est **indépendante**

...elle peut être **évaluée à part** en une seule fois

…elle peut être **évaluée hors** de la requête externe

Imbrication simple (2)

```
SELECT f FROM Fourniture
WHERE p IN ( SELECT p
             FROM Produit
             WHERE couleur = 'vert' ) ;
```

Imbrication simple (3)

- Analogie

```
T = T + 0 ;
for I = 1, n do {
    S = 0 ;
    for J = 1, m do { S = S + A[J] } ;
    T = T + S } ;
```

Imbrication corrélée (1)

- La requête interne **fait référence** à des éléments de la requête externe...

...elle en est **dépendante**

...elle **doit être évaluée répétitivement** dans la requête externe

Imbrication corrélée (2)

```
SELECT f FROM Fourniture AS f1
WHERE f IN
      ( SELECT f FROM Fourniture AS f2
        WHERE f1.f = f2.f AND f1.p <> f2.p ) ;
```

Imbrication corrélée (3)

- Analogie

```
T = T + 0 ;
for I = 1, n do {
    S = 0 ;
    for J = 1, m do { S = S + A[I,J] } ;
    T = T + S } ;
```

Imbrication corrélée (4)

- EXISTS n'a de sens que corrélée
- Exemple

```
SELECT f FROM Fourniture AS f1
WHERE EXISTS
        ( SELECT * FROM Fourniture AS f2
          WHERE f1.f = f2.f AND f1.p <> f2.p ) ;
```

Simulation de la division

Problématique

- La division relationnelle est **commode**, mais **pas définie** en SQL...

...la simuler !

Exemple division (1)

- Les numéros d'adhérents

  SELECT NumAdherent

  FROM Adherent

- Les numéros des adhérents qui pratiquent une activité

  SELECT NumAdherent

  FROM AdherentActivite

- Les numéros des adhérents qui pratiquent toutes les activités

  ???

Exemple division (2)

- Les numéros des adhérents qui pratiquent toutes les activités

SELECT NumAdherent, NumActivite

FROM AdherentActivite

/

SELECT NumActivite

FROM AdherentActivite

Méthode du groupement

- Rechercher tous les **NumAdherent** en relation avec tous les **NumActivite**

```
SELECT NumAdherent
FROM AdherentActivite
GROUP BY NumAdherent
HAVING COUNT(*)
            = (SELECT COUNT(NumActivite)
               FROM AdherentActivite)
```

- Ces adhérents sont en relation avec autant d'activités qu'il y a d'activités

Méthode des soustractions

- Rechercher tous les **NumAdherent** en relation avec tous les **NumActivite**

  R1 = SELECT NumAdherent FROM Adherent

  R2 = SELECT NumActivite FROM AdherentActivite

  R3 = SELECT x, y FROM R1, R2

  R4 = SELECT x FROM (R3 MINUS AdherentActivite) m

  R5 = R1 MINUS R4

- Ces adhérents ne forment pas de combinaisons nouvelles quand on les combine avec toutes les activités

Méthode du EXISTS

- Rechercher tous les **NumAdherent** en relation avec tous les **NumActivite**

```
SELECT NumAdherent FROM Adherent ad
WHERE NOT EXISTS
(SELECT * FROM AdherentActivite adac1
WHERE NOT EXISTS
(SELECT * FROM AdherentActivite adac2
WHERE adac2.NumActivite = adac1.NumActivite
AND adac2.NumAdherent = ad.NumAdherent))
```

- Pour chacun de ces adhérents, il n'existe pas d'activité avec qui il ne soit pas en relation

Conclusion

- Imbriquer les requêtes est normal...

...en particulier pour simuler la division

- Distinguer table, ligne, colonne, valeur, booléen