Rappel : Requêtes Conjonctives

Sous ensemble de SQL.

Les plus utilisées en pratique.

Celles qu'on sait très bien optimiser.

On a vu:

- Déclaratif: CQ $\subsetneq$ FO (pour faire de la belle théorie)
- Procédural : SRPJ $\subsetneq$ ALG (pour implanter, optimiser etc.)
- Utilisateur : SFW $\subsetneq$ SQL (ce qu'on utilise en pratique)

L'algèbre SRPJ

Manipule des relations (tables) via leur nom et celui de leurs attributs.

Salle(nom,horaire,titre)
Film(titre,réalisateur,acteur)

Contient quatre opérations:

σ: sélection, sélectionne des lignes suivant un critère donné.

π: projection, sélectionne des colonnes

$\bowtie$: jointure, fait un "pont" entre deux tables

ρ: renommage

Elle est très facile à implanter.

CQ

Langage déclaratif (il faudra le compiler)

C'est facile d'y exprimer ce que l'on veux.

Manipule des noms de relation sans se préoccuper du nom des attributs mais uniquement de leur ordre.

Film(x,y,z)

Grâce au schéma de relation on sait que x est in titre, y est un réalisateur et z un acteur.

La quantification porte sur le domaine actif : **adom**.

On se restreint aux requêtes sûres : celles où l'on peut associer (au moins) un nom d'attribut à chaque variable.

SFW

C'est SQL sans

GROUP-BY/HAVING

OR, UNION, EXCEPT

requêtes imbriquées: IN, NOT IN, EXISTS NOT EXISTS

Mais on garde

SELECT, FROM, WHERE, AND

Théorème d'équivalence:

SFW=SCQ=SRPJ

Le passage SFW $\rightarrow$ SRPJ correspond à la compilation de la requête. On reverra cela en détail plus tard.

Exemples

Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)

Tout sur le film 'Vertigo'

Film(x,y,z) $\wedge$ x='Vertigo' $\sigma_{\text{titre="Vertigo"}}(\text{Film})$

```
SELECT *
FROM Film
WHERE titre='Vertigo'
```

Les films qui ont un acteur qui est aussi le réalisateur

Film(x,y,z) $\wedge$ y=z $\sigma_{\text{réalisateur=acteur}}(\text{Film})$

```
SELECT *
FROM Film
WHERE acteur=réalisateur
```


Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)

La liste des salles de cinéma

$\exists\, y, z$ Salle(x,y,z) $\pi_{\text{nom}}(\text{Salle})$

```
SELECT nom
FROM Salle
```

Les acteurs du film 'Vertigo'

$\exists\, x, y$ (Film(x,y,z) $\wedge$ x='Vertigo') $\pi_{\text{acteur}}(\sigma_{\text{titre='Vertigo'}}(\text{Film}))$

```
SELECT acteur
FROM Film
WHERE titre='Vertigo'
```


Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)

Les salles où l'on joue un film avec Stewart

$\exists\, x, y, z, v$ (Film(x,y,z) $\wedge$ Salle(u,v,x) $\wedge$ z='Stewart')

$\pi_{\text{nom}}(\sigma_{\text{acteur='Stewart'}}(\text{Film} \bowtie \text{Salle}))$

```
SELECT nom
FROM Salle,Film
WHERE acteur='Stewart'
AND   Film.titre=Salle.titre
```

Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)
Produit(producteur,titre)

Quels sont les acteurs qui produisent un film dans lequel ils jouent?

$$\exists\, x, y \ \ (\text{Film(x,y,z)} \ \wedge \ \text{Produit(z,x)})$$

$$\pi_{\text{acteur}}(\sigma_{\text{producteur=acteur}}(\text{Produit} \bowtie \text{Film}))$$

```
SELECT acteur
FROM Film,Produit
WHERE Produit.titre=Film.titre
AND   producteur=acteur
```


Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)
Produit(producteur,titre)

Quels sont les acteurs qui produisent un film qu'ils ont réalisé?

$\exists\, x, z$ (Film(x,y,z) $\wedge$ Produit(y,x) $\wedge$ $\exists\, u, v$ Film(u,v,y))

$(\rho_{\text{réalisateur}\to\text{acteur}}\ (q) \bowtie \pi_{\text{acteur}}(\text{Film}))$

où $q \equiv \pi_{\text{réalisateur}}(\rho_{\text{producteur}\to\text{réalisateur}}(\text{Produit}) \bowtie \text{Film})$

```
SELECT F2.acteur
FROM Film F1, Produit, Film F2
WHERE F1.titre=Produit.titre
AND   F1.réalisateur=producteur
AND   F2.acteur=producteur
```


Aujourd'hui: Extensions

Ajout de la négation et de l'union.

Monotonicité des requêtes conjonctives.

Bases de la réécriture et du pouvoir d'expression.

Une opération redondante (mais utile) : ×

× , produit cartésien, opérateur binaire.

$$R_1 \times R_2$$

R_1 et R_2 sont des noms de relation

R_1 et R_2 n'ont pas d'attributs en commun

R

A	B
1	2
4	2
6	6
7	9

S

C	D
2	3
3	5

$R \times S$

A	B	C	D
1	2	2	3
1	2	3	5
4	2	2	3
4	2	3	5
6	6	2	3
6	6	3	5
7	9	2	3
7	9	3	5

Une autre opération redondante (mais utile) : ∩

∩ , intersection, opérateur binaire.

$$R_1 \cap R_2$$

R_1 et R_2 sont des noms de relation

R_1 et R_2 ont exactement les mêmes attributs

Quels sont les acteurs qui produisent un film qu'ils ont réalisé?

$\exists\, x, z$ (Film(x,y,z) $\wedge$ Produit(y,x) $\wedge$ $\exists\, u, v$ Film(u,v,y))

$(\rho_{\text{réalisateur}\to\text{acteur}}\ (q) \cap \pi_{\text{acteur}}(\text{Film}))$

où $q \equiv \pi_{\text{réalisateur}}(\rho_{\text{producteur}\to\text{réalisateur}}(\text{Produit}) \bowtie \text{Film})$

On rajoute l'union

- L'algèbre devient: SRPJU

  On rajoute un opérateur binaire d'union: ∪

  (comme pour ∩ les deux relations ont exactement les mêmes attributs)

  On autorise le "ou" (∨) dans le filtre d'une sélection

- Le calcul devient: pos-FO

  Idem que pour CQ mais la clôture se fait maintenant aussi avec l'opérateur ∨.

- Le langage utilisateur est: pos-SQL

  On autorise maintenant les opérateurs UNION et OR

Exemple

Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)

Où peut on voir 'Annie Hall' ou 'Manhattan'?

$\exists\, y, z$ (Salle(x,y,z) $\wedge$ (z="Annie Hall" $\vee$ z='Manhattan'))

$\pi_{\text{nom}}(\sigma_{\text{titre='Annie Hall'}\vee\text{titre='Manhattan'}}(\text{Salle}))$

$\pi_{\text{nom}}(\sigma_{\text{titre='Annie Hall'}}(\text{Salle}) \cup \sigma_{\text{titre='Manhattan'}}(\text{Salle}))$

```
SELECT nom
FROM Salle
WHERE titre='Annie Hall'
OR    titre='Manhattan'
```

```
SELECT nom
FROM Salle
WHERE titre='Annie Hall'
    UNION
SELECT nom
FROM Salle
WHERE titre='Manhattan'
```


Théorème d'équivalence:

pos-SFW=pos-FO=SRPJU

On rajoute la négation

- L'algèbre devient: ALG
  On rajoute un opérateur binaire de différence: $-$
  (comme pour $\cap$ et $\cup$ les deux relations ont exactement les mêmes attributs)
- Le calcul devient: CALC ou FO
  La clôture se fait maintenant aussi avec les opérateurs $\forall x$ et $\neg$.
- Le langage utilisateur est: mini-SQL
  SQL avec l'imbrication: IN, NOT IN, EXISTS, NOT EXISTS, MINUS
  mais sans les agrégats: GROUP-BY, HAVING, AVG, SUM, MIN, MAX

Exemple

Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)
Produit(producteur,titre)

Quels films ne passent dans aucune salle ?

$(\exists\, r, a$ Film(t,r,a)) $\wedge$ $(\forall\, n, h$ $\neg$Salle(n,h,t))

$\pi_{\text{titre}}(\text{Film}) - \pi_{\text{titre}}(\text{Salle})$

```
SELECT titre
FROM Film
WHERE NOT EXISTS
  (SELECT nom
   FROM Salle
   WHERE Salle.titre=Film.titre)
```


Exemple

Salle(nom,horaire,titre) Film(titre,réalisateur,acteur)
Produit(producteur,titre)

Qui produit tous les films de Kurosawa?

$\forall\, t, r, a$ ((Film(t,r,a) $\wedge$ r="Kurosawa") $\rightarrow$ Produit(x,t))

$P - (\pi_{\text{producteur}}(((P \bowtie T) - \text{Produit}) \bowtie \sigma_{\text{réalisateur="Kurosawa"}}(\text{Film})))$

où P est $\pi_{\text{producteur}}(\text{Produit})$

T est $\pi_{\text{titre}}(\text{Film})$


```
SELECT P1.producteur
FROM Produit P1
WHERE NOT EXISTS
  (SELECT titre
   FROM Film
   WHERE réalisateur='Kurosama'
   AND NOT EXISTS
      (SELECT titre
       FROM   Produit P2
       WHERE  P2.producteur=P1.producteur
       AND    P2.titre=Film.titre))
```


Théorème d'équivalence:

mini-SQL = FO (CALC) = ALG

Les requêtes suivantes sont-elles conjonctives?

Quels acteurs jouent dans tous les films de Welles ?

Qui produit tous les films de Kurosawa ?

Quels films ne passent dans aucune salle ?

Qui produit un film qui ne passe dans aucune salle ?

Quels producteurs voient tous les films de Kurosawa ?

Non!

Une requête q est dite monotone si pour toute instances I et J

$$I \subseteq J \implies q(I) \subseteq q(J).$$

Théorème: Les requêtes conjonctives sont monotones

preuve: par induction sur la formule de CQ correspondante.

Les requêtes suivantes sont-elles conjonctives?

Quels acteurs jouent dans tous les films de Welles ?

Qui produit tous les films de Kurosawa ?

Quels films ne passent dans aucune salle ?

Qui produit un film qui ne passe dans aucune salle ?

Quels producteurs voient tous les films de Kurosawa ?

Non! car elles ne sont pas monotones

Équivalence algébrique

Commutativité et associativité de la jointure

$$E_1 \bowtie E_2 = E_2 \bowtie E_1,$$
$$(E_1 \bowtie E_2) \bowtie E_3 = E_1 \bowtie (E_2 \bowtie E_3).$$

Cascade de projections

$$\pi_{A_1,\dots,A_n}(\pi_{B_1,\dots,B_m}(E)) = \pi_{A_1,\dots,A_n}(E)$$

Cascade de sélections

$$\sigma_{F_1}(\sigma_{F_2}(E)) = \sigma_{F_1 \wedge F_2}(E)$$

Commutation sélection et projection

Si F ne porte que sur $A_1, \dots, A_n$,

$$\pi_{A_1,\dots,A_n}(\sigma_F(E)) = \sigma_F(\pi_{A_1,\dots,A_n}(E))$$

Si F porte aussi sur $B_1, \dots, B_m$,

$$\pi_{A_1,\dots,A_n}(\sigma_F(E)) = \pi_{A_1,\dots,A_n}(\sigma_F(\pi_{A_1,\dots,A_n,B_1,\dots,B_m}(E)))$$

Commutation sélection et $\cup$ $-$ $\bowtie$

$$\sigma_F(E_1\ OP\ E_2) = \sigma_F(E_1)\ OP\ \sigma_F(E_2)$$

Commutation projection et $\cup$

$$\pi_{A_1,\dots,A_n}(E_1 \cup E_2) = \pi_{A_1,\dots,A_n}(E_1) \cup \pi_{A_1,\dots,A_n}(E_2)$$

REQUÊTE : *Les salles où l'on joue un film avec Stewart*

$$\pi_{\text{nom}}(\sigma_{\text{acteur="Stewart"}}(\text{Film} \bowtie \text{Salle}))$$

est équivalent à:

$$\pi_{\text{nom}}(\sigma_{\text{acteur="Stewart"}}(\text{Film}) \bowtie \text{Salle})$$

Forme normale

Toute formule q de SRPJ peut s'écrire sous la forme:

$$\pi_A(\sigma_F(T_1 \times \cdots \times T_n))$$

où A est un ensemble de noms d'attribut,

F est un ensemble de prédicats de filtre

et pour tout i, T_i est soit R soit ρ_f(R) pour un certain renommage f.

Pouvoir d'expression

Quelles sont les requêtes qu'on ne va pas pouvoir exprimer dans CALC et donc dans ALG et SQL?

La clôture transitive: Soit le schéma de relation Fils(nomenfant,nomparent)
requête: quels sont les ancêtres de "John"?

Théorème: cette requête n'est pas exprimable dans CALC

Preuve: difficile!!

Exercices

Pour chacune des requêtes de la feuille de TD2:

Dire si elle est conjonctive ou non (justifier)

L'exprimer dans SQL, ALG et CALC