Bases de Données

Cyril Labbé

LIG, Université de Grenoble, France
first.last@imag.fr

http://membres-lig.imag.fr/labbe/M1BD/CoursM1BD_4.pdf

Table of Contents

Traitement et optimisation de requêtes

SQL est un language déclaratif :

- spécifie les propriétés des données qu'il souhaite
- ne spécifie par comment les obtenir (i.e pas d'algorithme)

$\Longrightarrow$ Le SGBD à la charge de choisir la meilleure stratégie pour une évaluation optimale des requêtes.

Deux grandes classes de techniques d'optimisation de requêtes :

- suivre des règles heuristiques sur l'ordre des opérations dans une requête
- approche à base de coût pour une selection optimale des chemins d'accès. Comparer différentes stratégies en fonction de leurs coût et choisir celle qui minimise le coût.

En général l'accès au support de stockage constitue le coût dominant.
L'objectif est donc de réduire les E/S

Table of Contents

Exemple

...

```
1  Select * from
2   Sells S join Beers B on (BeerName)
3  where
4   B.BrewerName='FFB' and S.price>4.5
```

Cardinalités

- Cardinalité de Sells = 1000
- Cardinalité de Beers = 50
- 5 bières brassées par 'FFB'
- 100 bières vendues à plus de 4.5

Plan 1 : $\sigma_{(BrewerName='FFB' \wedge price>4.5 \wedge Sells.BeerName=Beers.BeerName)}(Sells \times Beers)$

Lecture de Sells (1000), lecture de Beers (50), nombre de tuples du produit cartésiens (50*1000)
$\Longrightarrow$ 101050 E/S

Plan 2 : $\sigma_{(BrewerName='FFB' \wedge price>4.5)}(Sells \bowtie Beers)$

Lecture de Sells (1000), lecture de Beers (50), nombre de tuples de la jointure (1000)
$\Longrightarrow$ 3050 E/S

Plan 3 : $\sigma_{(price>4.5)} Sells \bowtie \sigma_{(BrewerName='FFB')} Beers$

Lecture de Sells (1000), lecture de Beers (50), avant jointure 100+5 tuples, après jointure 100+5
$\Longrightarrow$ 1050+210 = 1260

Décomposition d'une requête

Objectif : transformer une requête SQL en une requête en alg relationnelle.

- vérifier qu'elle est correcte (syntaxique / sémantique)
- décomposer en *opérations* élémantaires
- choisir le meilleure plan d'execution

Les étapes :

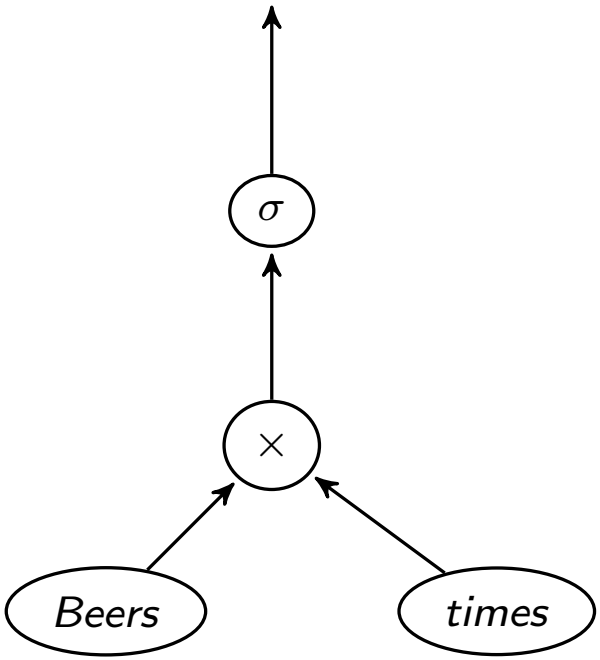
- analyse
- normalisation
- analyse sémantique
- simplification
- restructuration (optimisation)

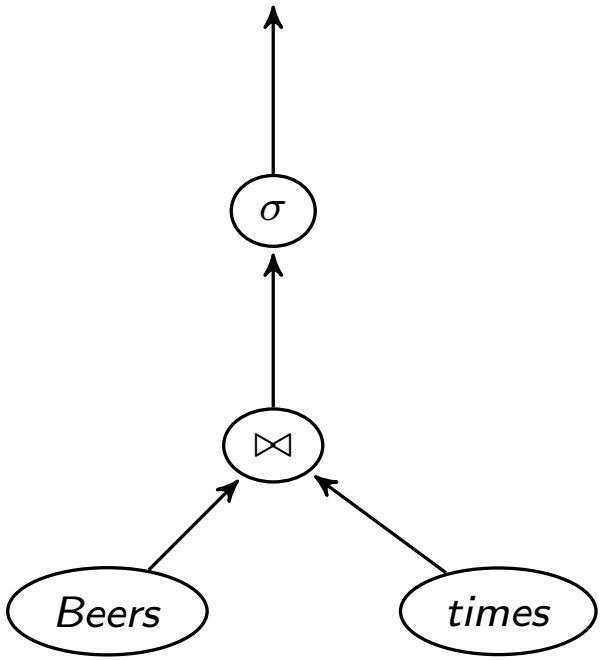
Analyse

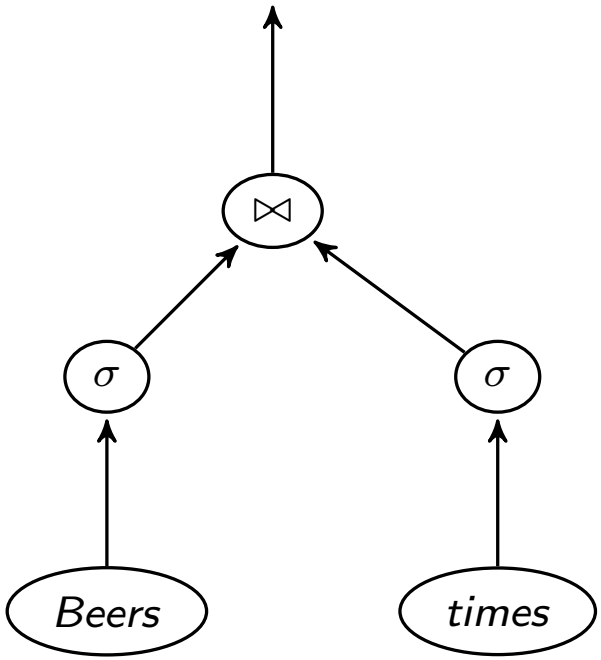
Technique de compilation (cf AS))

A la fin de l'étape la requête est transformée en une représentation interne mieux adaptée. En générale sous forme d'un arbre

- Un noeud feuille pour chaque relation de la requête
- Un noeud interne pour chaque opération d'algèbre relationnelle
- La racine de l'arbre représente le résultat de la requête







Normalisation/Sémantique/Simplification

Normalisation

de la forme des prédicats :

- forme conjonctive
- forme disjonctive

Exemple :

- $(BeerName =' no' \vee price > 3) \wedge (BrewerName =' bno')$
- $(BeerName =' no' \wedge BrewerName =' bno') \vee$
  $(price > 3 \wedge (BrewerName =' bno')$

Analyse sémantique

Rejeter les requêtes normalisées incorrectes : $(price < 5.6 \wedge price > 10)$
Problème de la satisfiabilité... (SAT)

Simplification

Détecter et éliminer des sous-requêtes communes : transformer la requête en une requête plus simple sémantiquement équivalente.

Simplification : exemple

Création d'une vue:

```
1  Create view MonBar
2  select BeerName, price
3  from Sells natural join Beers
4  where BarName='t' ;
```

Requête utilisateur

```
1  select * from MonBar where BarName='t' or BarName='b2'
```

Requête à simplifier

```
1  select * from
2     (select BeerName, price
3      from Sells natural join Beers where BarName='t')
4    where BarName='t' or BarName='b2'
```

Restructuration/optimisation

Restructuration/optimisation :

Choisir la structuration et l'implantation optimal.

- approche par Heuristique / Optimisation algébrique
- approche à base de coût

Table of Contents

Règles de tranformation

Règles algébriques

Utiliser les règles de transformation des opérateurs de l'algèbre relationnelle pour obtenir des *bonnes* (à défaut d'optimales) stratégie d'execution.

Groupement/dégroupement de la sélection

Une sélection avec une condition de selection conjonctive se décompose en des opérations de sélection

$\sigma_{p \wedge q \wedge r}(R) = \sigma_p(\sigma_q(\sigma_r(R)))$

Commutativité de la sélection

$\sigma_p(\sigma_q(R)) = \sigma_q(\sigma_p(R))$

Fusion de la projections

Si $I \subset m \subset ... \subset n$ alors

$\pi_I \pi_m ... \pi_n(R) = \pi_I(R)$

Quasi-commutativité de la sélection et de la projection

Soit *cds* une condition de selection impliquant $A_1, A_2, ...A_n$ (noté $cds \in \{A_i\}_{i=1..n}$) alors :

$\pi_{A_1} \pi_{A_2} ... \pi_{A_n}(\sigma_{cds}(R)) = \sigma_{cds}(\pi_{A_1} \pi_{A_2} ... \pi_{A_n}(R))$

Règles de tranformation

Commutativité θ-jointure

$R \bowtie_\theta S = S \bowtie_\theta R$
$R \bowtie S = S \bowtie R$
$R \times S = S \times R$

Quasi-commutativité sélection et θ-jointure

Si le prédicat de selection *cds* n'implique que des attributs d'une des relations alors la selection et la θ-jointure sont commutatives (e.g. $cds \in Att(R)$):

$$\sigma_{cds}(R \bowtie_\theta S) = S \bowtie_\theta (\sigma_{cds} R)$$

Si le prédicat de selection *cds* est de la forme $p \wedge q$ où $p \in Att(R)$ et $q \in Att(S)$ alors :

$$\sigma_{p \wedge q}(R \bowtie_\theta S) = (\sigma_p R) \bowtie_\theta (\sigma_q S)$$

Quasi-commutativité projection et θ-jointure

Soit L la liste de projection de la forme $L = L_1 \cup L_2$ où $L_1 \in Att(R)$ et $L_2 \in Att(S)$
Soit θ une θ condition n'impliquant que des attribut de L alors :

$$\pi_L(R \bowtie_\theta S) = (\pi_{L_1} R) \bowtie_\theta (\pi_{L_2} S)$$

Règles de tranformation

Commutativité union et intersection

$$R \cup S = S \cup R$$

$$R \cap S = S \cap R$$

Distributivité selection et opération sur ensemble

$$\sigma_p(R \cup S) = (\sigma_p S) \cup (\sigma_p R)$$

$$\sigma_p(R \cap S) = (\sigma_p S) \cap (\sigma_p R)$$

$$\sigma_p(R - S) = (\sigma_p R) - (\sigma_p S)$$

Commutativité projection et union

$$\pi_L(R \cup S) = (\pi_L S) \cup (\pi_L R)$$

Associativité de l'union et de l'intersection

$$(R \cup S) \cup T = R \cup (S \cup T)$$

$$(R \cap S) \cap T = R \cap (S \cap T)$$

Associativité de la θ jointure

$$(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$$

$$(R \times S) \times T = R \times (S \times T)$$

Si $q \in Att(S) \cup Att(T), q \notin Att(R)$ et $r \in Att(R) \cup Att(T), r \notin Att(S)$ alors

$$(R \bowtie_p S) \bowtie_{(q \wedge r)} T = R \bowtie_{(p \wedge r)} (S \bowtie_q T)$$

Heuristiques

Objectif

Réduire au plus vite la taille des résultats intermédiaires

Règles

1. Appliquer les opérations de sélection le plus tôt possible.
   - la selection réduit la cardinalité des tables
   - déplacer les sélections le plus bas possible dans l'arbre
2. Combiner produit cartésien avec la selection pour obtenir une jointure

$$\sigma_{R.a\theta S.b}(R \times S) = R \bowtie_{R.a\theta S.b} S$$

3. Utiliser l'associativité des opérateurs binaires pour que les opérations les plus restrictives s'exécutent en premier :

$$R \bowtie_{R.a\theta S.b} S \bowtie_{S.c\theta T.d} T$$

   Executer en premier la jointure qui donne le plus petit résultat.
4. Appliquer les opérations de projection le plus tôt possible pour réduire la taille des tuples (! pas forcement intéressant en pratique !)
5. Ne calculer qu'une fois les expression qui apparaissent plusieurs fois.

Exemple

BD Beers

- *Beers(BeerName, BrewerName, Country)*
- *Drinkers(DrinkName, Drink@, DrinkPhone)*
- *Bars(BarName, Bar@, licence, OpenDate)*
- *Sells(BeerName#,BarName#, price)*
- *Frenquents(BarName#,DrinkName#)*
- *Drinks(BeerName#,DrinkName#)*

```
1  Select DrinkName, BrewerName
2  From Beers, Drinks, Drinkers, Sells
3  Where (Drink@ Like '%grenoble%')
4        and Country='Australia'
5        and price > 5
6        and Beers.BeerName = Drinks.BeerName
7        and Drinks.BeerName = Sells.BeerName
8        and Drinkers.DrinkName = Drinks.DrinkName
```

Bibliographie.[Ullman and Widom, 2001, Connolly and Begg, 2004, Garcia-Molina et al., 2008, Rigaux, 2001]

Connolly, T. M. and Begg, C. E. (2004).

Database Systems: A Practical Approach to Design, Implementation and Management (4th Edition).
Pearson Addison Wesley.

Garcia-Molina, H., Ullman, J. D., and Widom, J. (2008).

Database Systems: The Complete Book.
Prentice Hall Press, Upper Saddle River, NJ, USA, 2 edition.

Rigaux, P. (2001).

Cours de bases de données.
Technical report, CNAM.

Ullman, J. D. and Widom, J. (2001).

A First Course in Database Systems.
Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition.