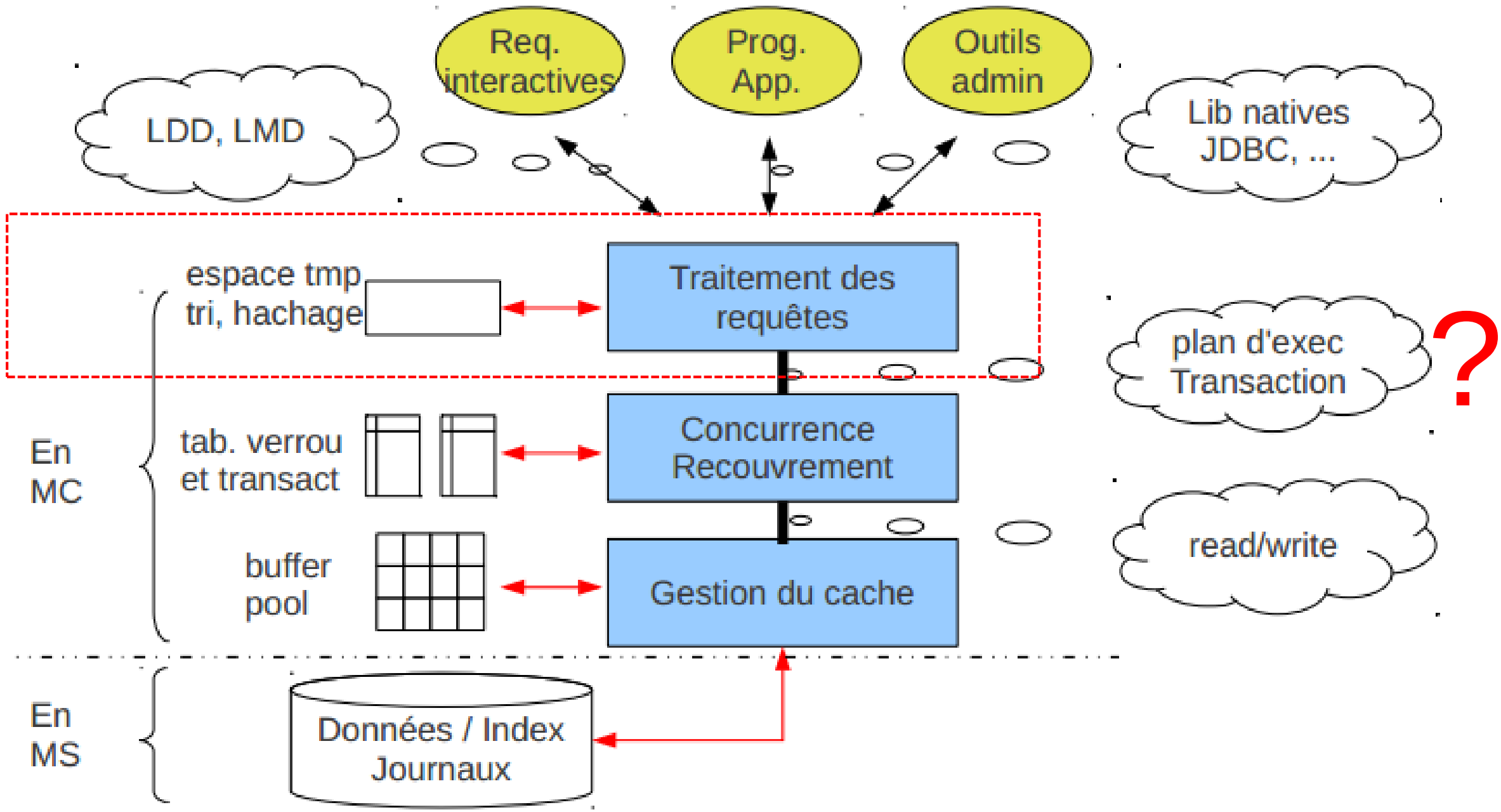
Cours 3

Traitement des Requêtes SQL

BDA
Master GL- 2019- 2020

OUARED Abdelkader

Objectif de Cours

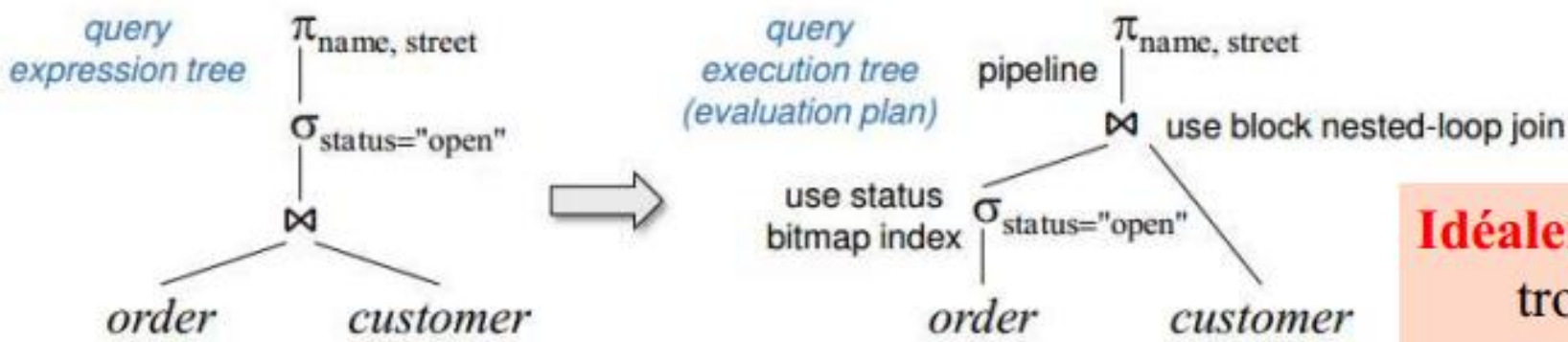


Optimisation des requêtes

- Plans d'exécutions équivalents
  - Transformer la requête SQL pour le plan de requête suivant

```
SELECT name, street
FROM Customer, Order
WHERE Order.customerID = Customer.customerID AND status = 'open';
```

$$\pi_{\text{name, street}} (\sigma_{\text{status="open"}} (\text{order} \bowtie \text{customer}))$$



Idéalement :
trouver le meilleur plan.
Pratiquement :
Éviter le pire plans!

Processus d'exécution d'une requête SQL

- Requête → Arbre algébrique
- Arbre algébrique → plusieurs plans d 'exécutions
  n operateur → n ! Plan d'exécution
- Comment choisir le plan optimum → Pb combinatoire (NP)!!

Optimisation des requêtes

- Deux grandes techniques d'optimisation
  - Utilisation de règles heuristiques (Rule-Based)
    - Ré-ordonne les opérations dans les arbres de requêtes
  - Évaluation systématique des coûts (Cost-Based)
    - Choix du plan d'exécution le moin coûteux

- **Un optimiseur de requête combine généralement ces deux techniques**

Optimisation des requêtes à base de règles

- Principale règles de Re-structuration
- Règles d’équivalences de l ’algèbre relationnelle
- Heuristique d'Optimisation

Optimisation des requêtes

Rule-based Optimization Approach (RA)

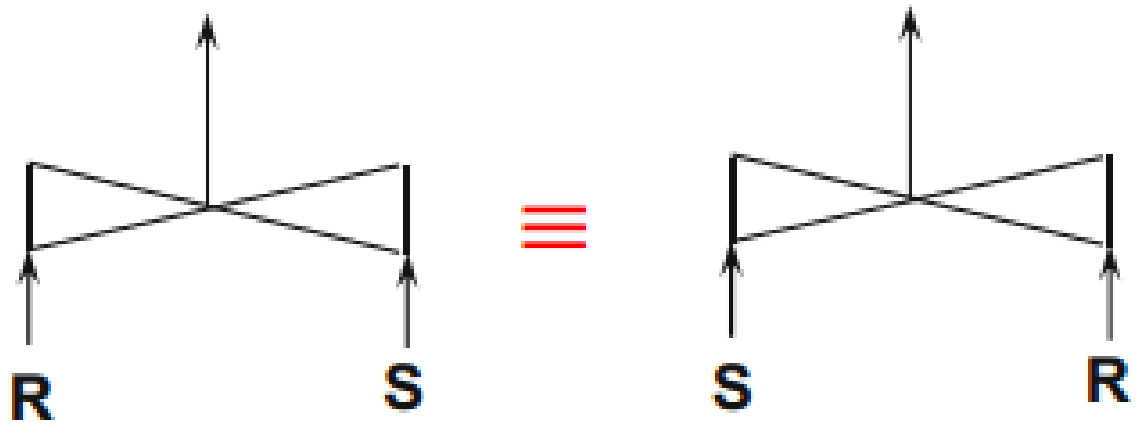
- **Problème**
  - –suivant l'ordre des opérateurs algébriques dans un arbre, le coût d'exécution est diffèrent
- **Pourquoi?**
  - –1. le coût des opérateurs varient en fonction du volume des données traitées

    i.e., plus le nombre de tuple des relations traitées est petit, plus les coûts CPU et d'E/S sont minimisés
  - –2. certains opérateurs diminuent le volume des données comme restriction et projection

Règles d’équivalences de l ’algèbre relationnelle (Règles de Restructuration)

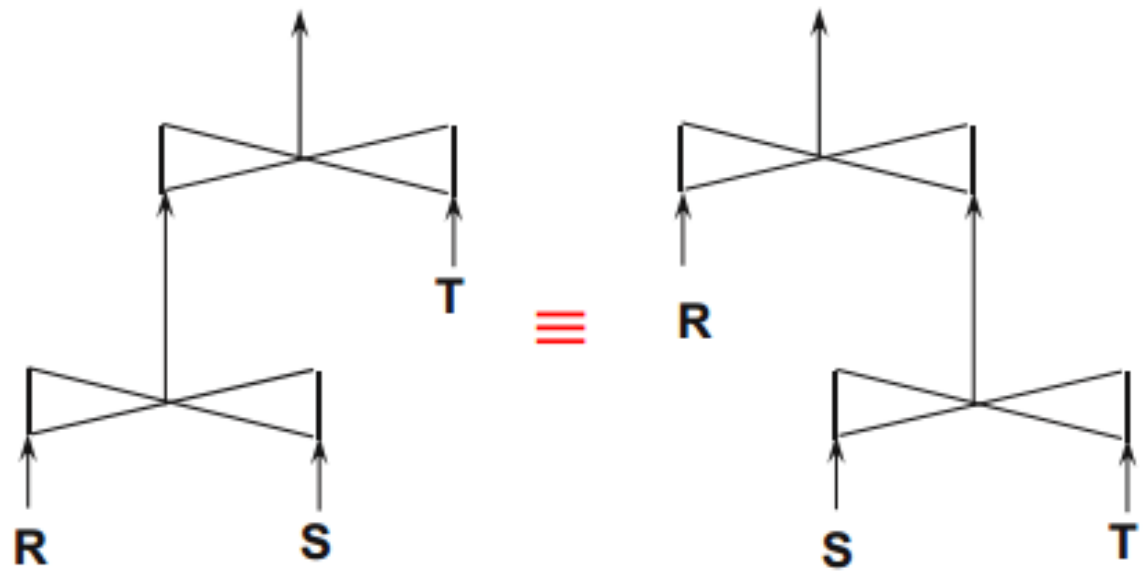
Principale règles de Re-structuration

- Commutativité des jointures



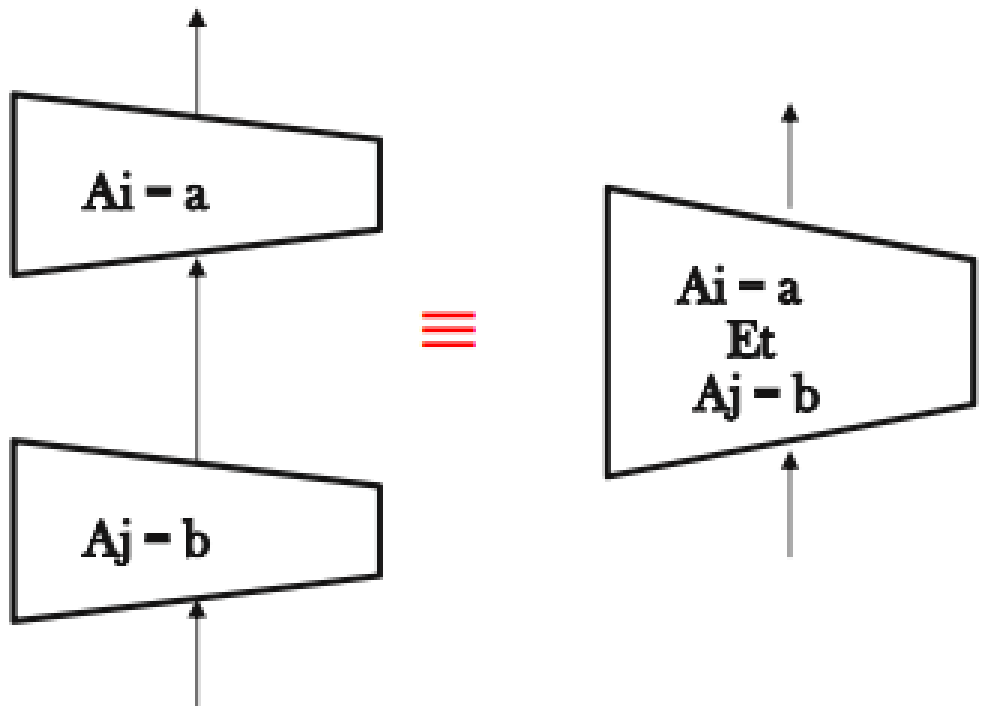
Principale règles de Restructuration

- Associativité des jointures
  - –Il existe N!/2 arbre de jointure de N relations.
  - –Parmi les jointures, certaines sont des produits cartésiens.



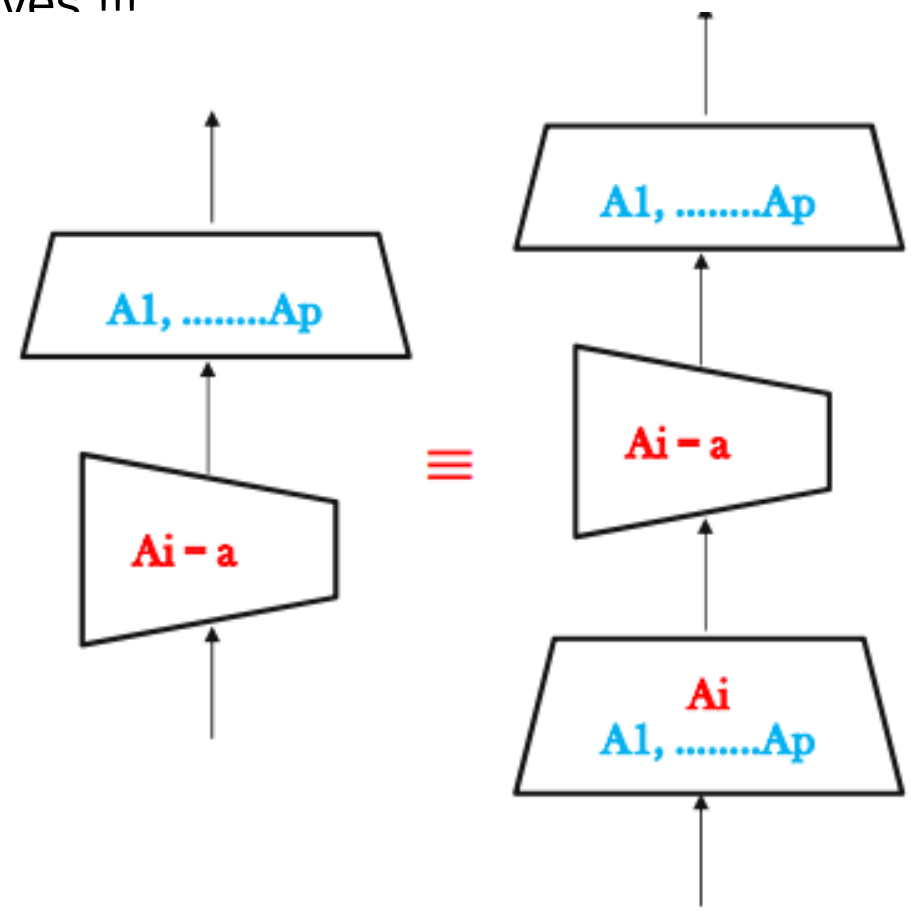
Principale règles de Restructuration

- Groupage des restrictions



Optimisation des requêtes à base de règles

- **Principale règles de Restructuration**
  - Semi-commutativité des projections et restrictions
    - Il est possible de descendre les projections, mais les attributs utilisés dans la suite doivent être conservés !!!



RBO: Rule-based optimizer

Règles d’équivalences de l ’algèbre relationnelle

1. Eclatement d'une sélection conjonctive (SE)
   1. $\sigma_{e1 \text{ ET } e2}(T) =$
2. Commutativité de la sélection (SC)
   1. $\sigma_{e1}(\sigma_{e2}(T)) =$
3. Elimination des projections en cascades (PE)
   1. $\pi_{liste1}(\pi_{liste2}(T)) =$
4. Inversion σ et π
   1. $\pi_{A1, \ldots, An}(\sigma_{F}(E)) =$
   2. $\pi_{A1, \ldots, An}(\sigma_{F}(E)) =$

Optimisation des requêtes à base de règles

Règles d'équivalences de l 'algèbre relationnelle

5. Commutativité de la jointure (JC)
   1. $T_1 \bowtie T_2 =$
6. Associativité de la jointure (JA)
   1. $T_1 \bowtie (T_2 \bowtie T_3) =$
7. Commutativité restreinte de la sélection et de la jointure (CSJ)
   - $\sigma_e (T_1 \blacktriangleright T_2) =$
     - si l'expression de sélection e ne contient que des colonnes de T_1
8. Commutativité restreinte de la projection et de la sélection (CPS)
   - $\pi_{listeColonnes} (\sigma_e (T)) =$
9. Commutativité restreinte de la projection et de la jointure (CPJ)
   - $\pi_{listeColonnes} (T_1 \blacktriangleright T_2) =$

Principes d'optimisation des expressions algébriques

Optimisation des requêtes à base de règles

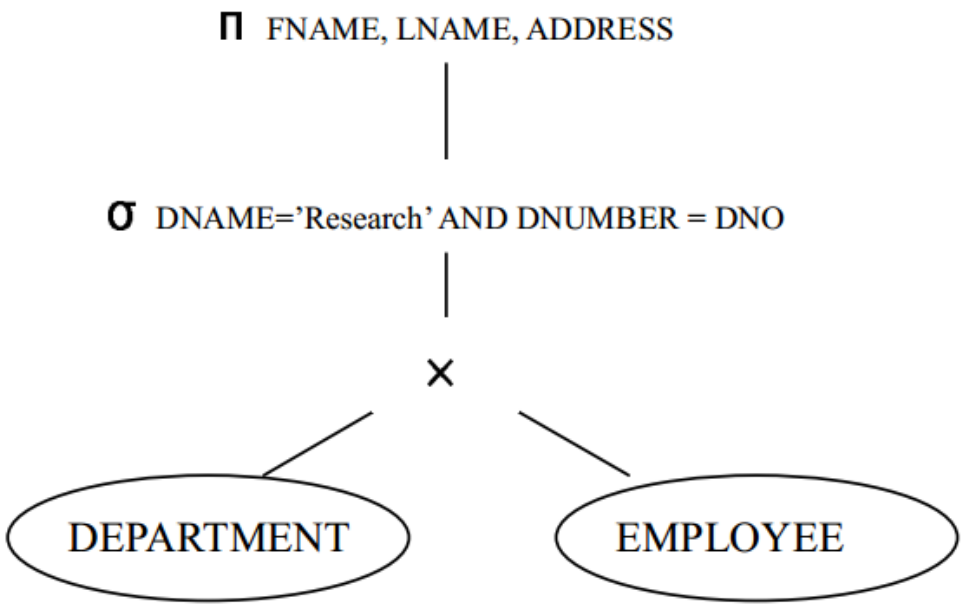
Principes d'optimisation des expressions algébriques (Heuristique d'Optimisation)

- Séparer chaque sélection $\sigma_{F1 \wedge \ldots \wedge Fn}(E)$ en une cascade $\sigma_{F1}(\sigma_{F2} \ldots (\sigma_{Fn}(E)))$
- Descendre chaque sélection leplus bas possible dans l'arbre algébrique
- Descendre chaque projection aussi bas possible dans l'arbre algébrique
- Descendre des cascades de sélection et de projection dans une sélection seule, une projection seule, ou une sélection suivie par une projection
- Regrouper les nœuds intérieurs de l'arbre autour des opérateurs binaires (*, -,∪). Chaque opérateur binaire crée un groupe
- Produire un programme comportant une étape pour chaque groupe, à évaluer dans n'importe quel ordre mais tant qu'aucun groupe ne soit évalué avant ses groupes descendants

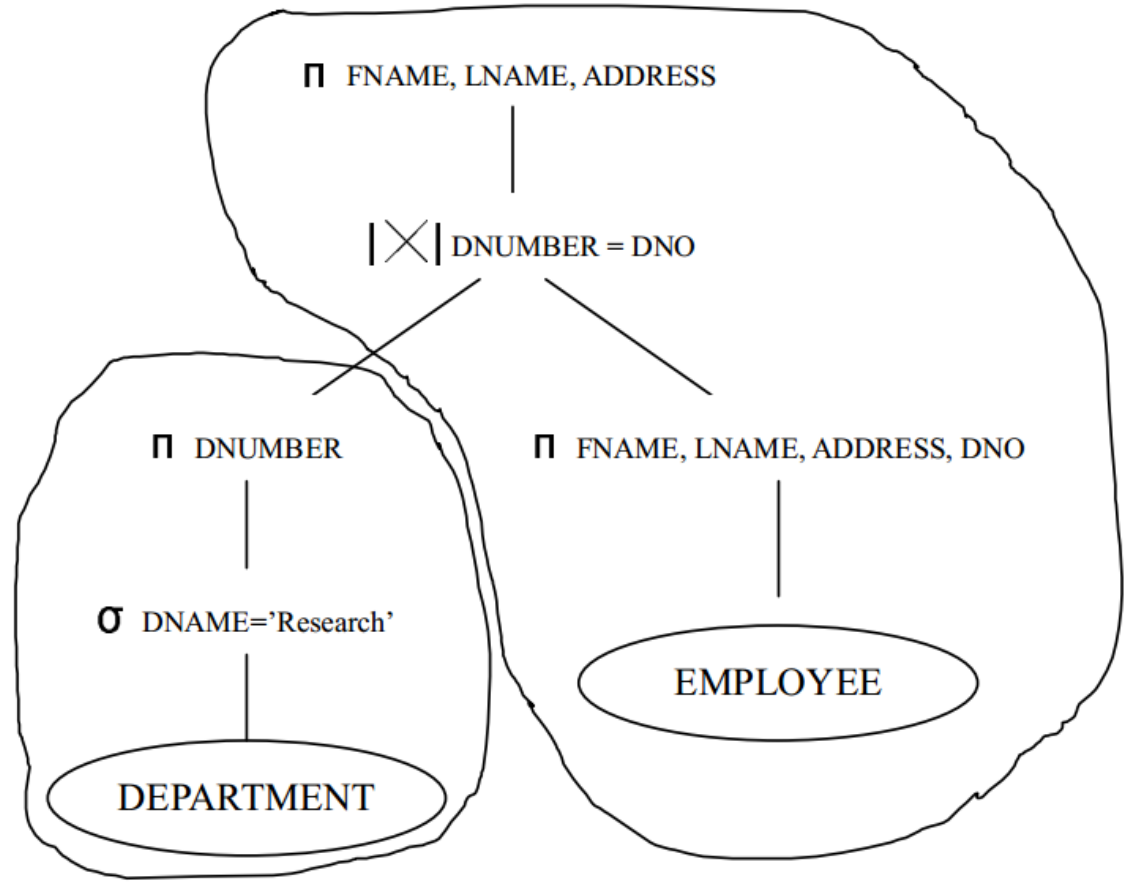
Exemple

Q1: SELECT FNAME, LNAME, ADDRESS
FROM EMPLOYEE, DEPARTMENT
WHERE DNAME = 'Research' AND DNUMBER = DNO

Initial tree:

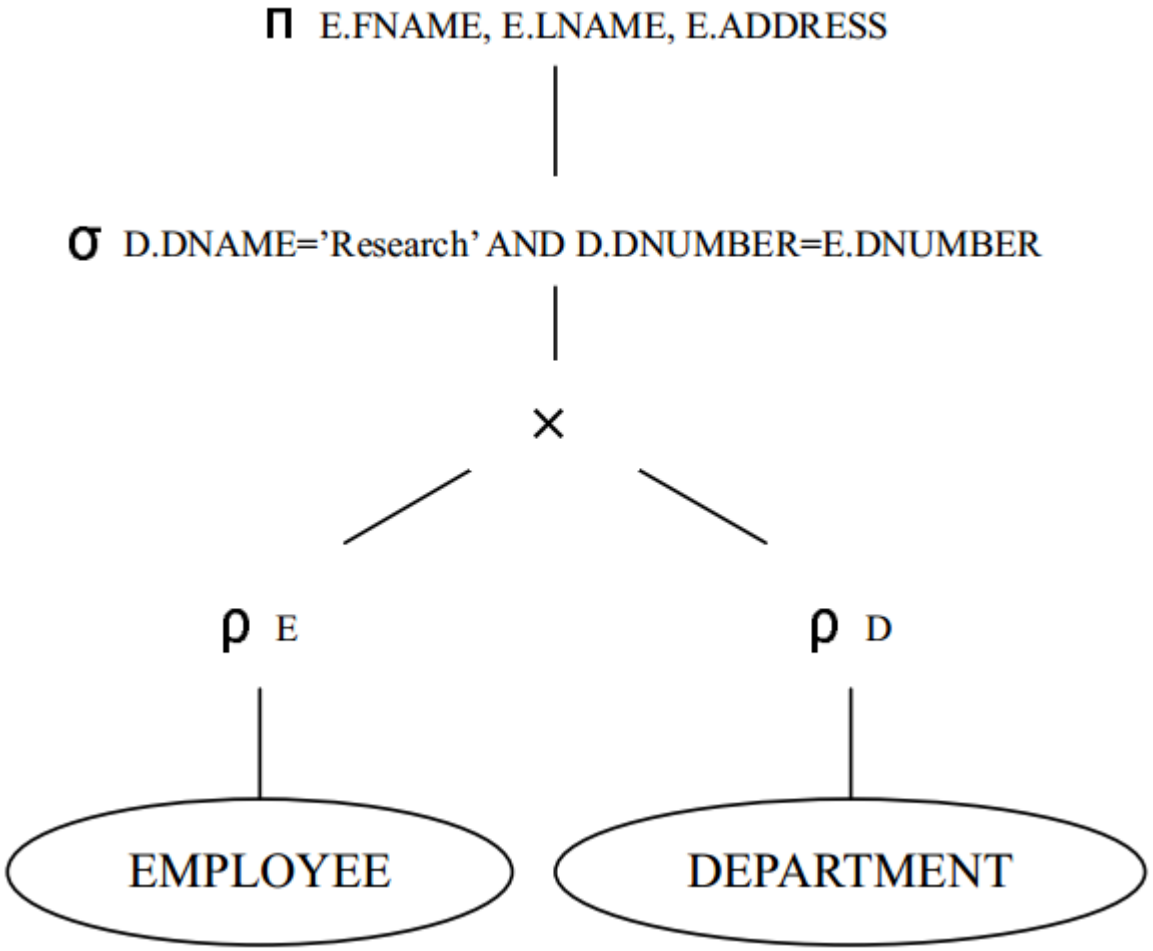


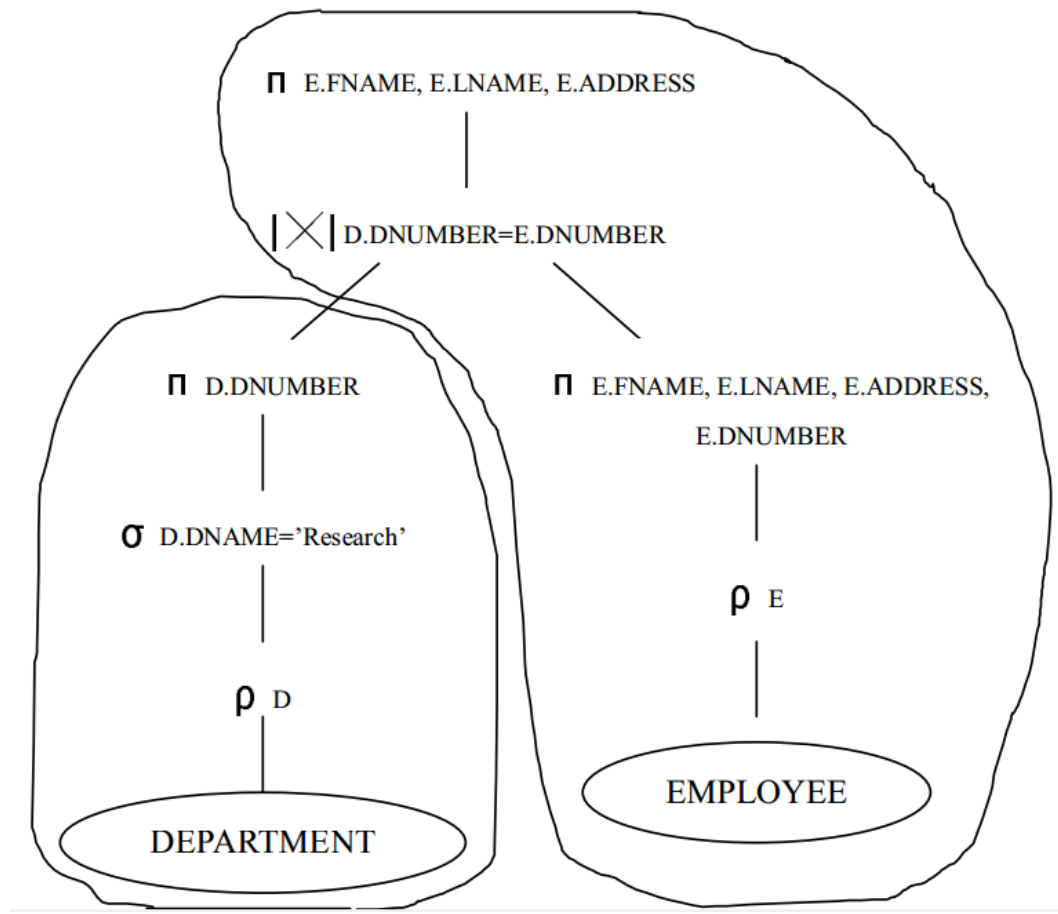
Optimized tree:



Exemple

Q1B: SELECT E.FNAME, E.NAME, E.ADDRESS
FROM EMPLOYEE E, DEPARTMENT D
WHERE D.NAME='Research' AND D.DNUMBER=E.DNUMBER



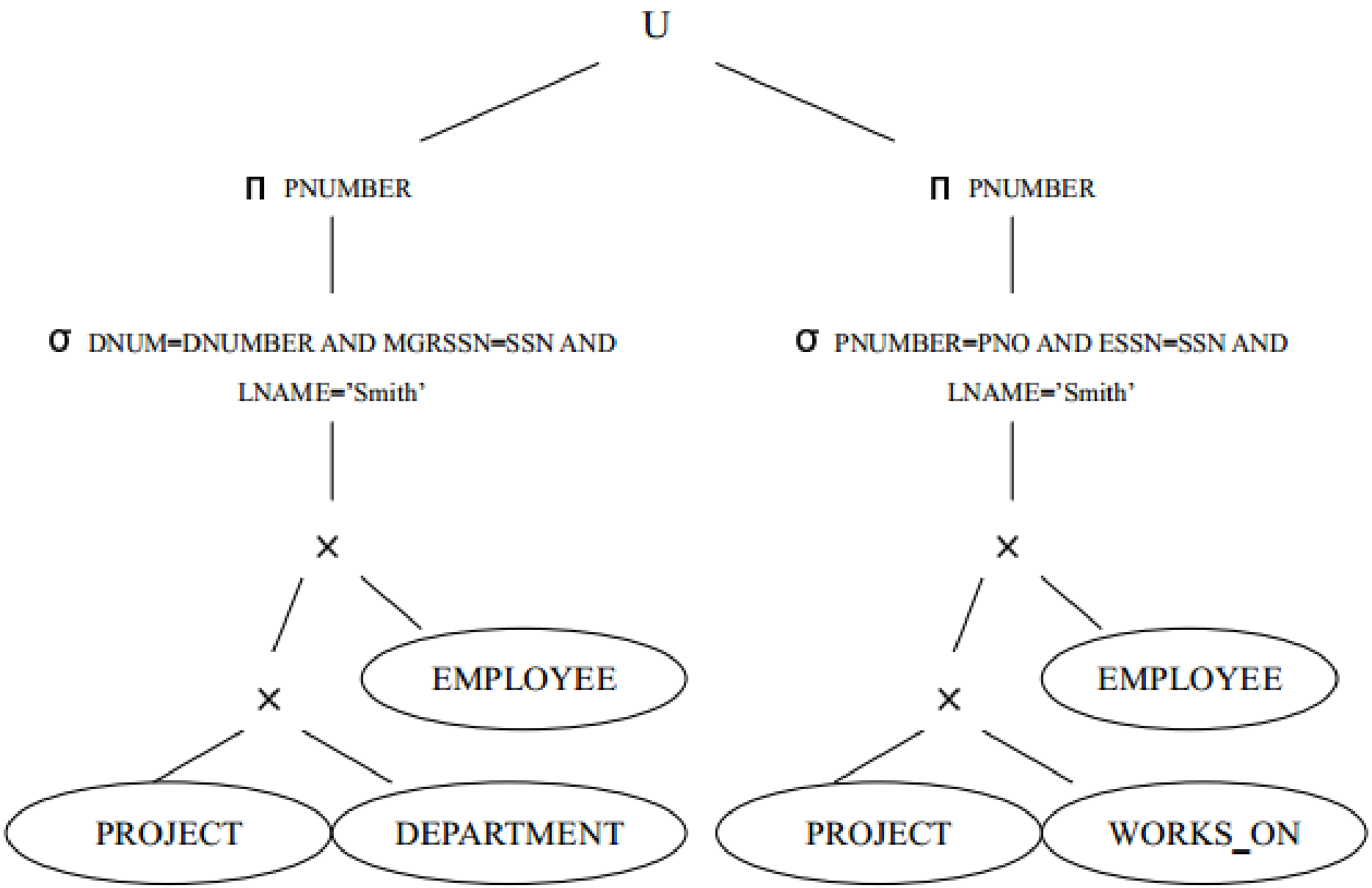


Exemple

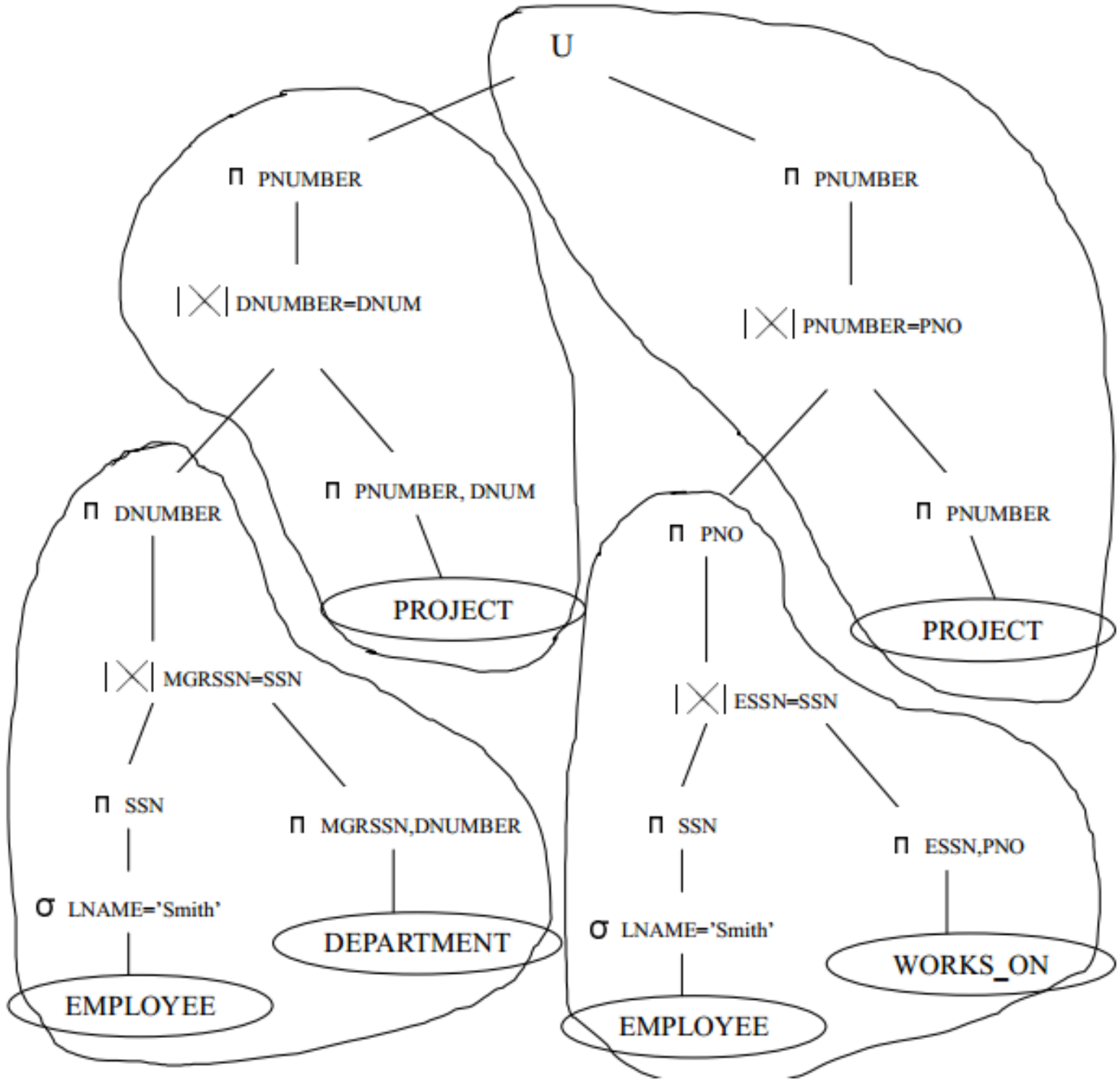
```
Q4:     (SELECT   DISTINCT PNUMBER
        FROM      PROJECT, DEPARTMENT, EMPLOYEE
        WHERE     DNUM=DNUMBER AND MGRSSN=SSN AND
                  LNAME='Smith')
        UNION
        (SELECT   DISTINCT PNUMBER
        FROM      PROJECT, WORKS_ON, EMPLOYEE
        WHERE     PNUMBER=PNO AND ESSN=SSN AND LNAME='Smith')
```

Exemple

Initial tree:



Exemple


U
π PNUMBER
|×| DNUMBER=DNUM
π PNUMBER, DNUM
π DNUMBER
PROJECT
|×| MGRSSN=SSN
π SSN
π MGRSSN,DNUMBER
σ LNAME='Smith'
DEPARTMENT
EMPLOYEE
π PNUMBER
|×| PNUMBER=PNO
π PNO
π PNUMBER
PROJECT
|×| ESSN=SSN
π SSN
π ESSN,PNO
σ LNAME='Smith'
WORKS_ON
EMPLOYEE

Optimisation des requêtes à base de règles

- Rule based optimisation
  - Pushdown select and project
  - Faire operation jointure

Exercice applicatif

- Soient trois relations: Livre, Prêt et Lecteur
  - Livre(ISBN, titre, auteur, éditeur)
  - Prêt(N°carteP, ISBNP, date)
  - Lecteur(N°carte, nom, adresse)
  - Liste des noms des lecteurs et des titres des livres pour tous les prêts effectués avant le 23/03/2015
  - Trouver le meilleur plan d'exécution de cette requête

Comment forcer l'optimiseur pour utiliser le RBO- cas de Oracle

hint

```
select /*+ RULE */ T1.COL1

from T1,T2

where T1.COL1=T2.COL1 and T1.COL1 ='5807'
```

Contrôle du processus d'optimisation-Cas Oracle

- Hints
  - Les directives‘’hints”sont des moyens déforcer l’optimiseur à choisir un plan d’exécution.
  - Les directives peuvent influencer les plans d’exécution et par conséquent la performance des requêtes

```
SELECT  /*+ some hints here)*/
```

- **Exemple**

```
SELECT  /*+ INDEX(EMPLOYÉ INDEX_SEXE)*/
FROM Employé,
WHERE Sexe_Emp= ‘F‘
```

Rewrite SQL

Remplacer

- NOT EXIST → NOT IN
- JOIN → EXIST
- EXIST → DISTINCT
- UNION → OR (Pourquoi?)
- WHERE → ORDER BY
- Having → remplacer par WHERE (Pourquoi?)
- Eviter les sous requêtes
- …
- **Exercice**

On considère la BD relationnelle suivante :

Student (IDS, GPA)

Plays (IDS, sport)

Et la requête **Q1** suivante :

```
SELECT sport
FROM Student, Plays
WHERE Student.ID=Plays.ID
GROUP BY sport
HAVING min(GPA) > 2.0
```

☞Ecrire une requête SQL équivalente de **Q1** sans utilisation de la clause "**HAVING**"

L’outil EXPLAIN

L’outil EXPLAIN donne le plan d’exécution d’une requête. La description comprend :

- Le chemin d’accès utilisé.
- Les opérations physiques (tri, fusion, intersection,...).
- L’ordre des opérations.
- Il est représentable par un arbre
- **EXPLAIN: exemple Oracle**

EXPLAIN PLAN SET

STATEMENT_ID=’JoinSelIndex’ FOR
SELECT R.idProduit
FROM Client C, Commande R
WHERE R.idClient = C.idClient
AND nom = ’Scott’;

Le résultat de EXPLAIN :

0 SELECT STATEMENT
1 NESTED LOOPS
2 TABLE ACCESS BY ROWID CLIENT
3 INDEX RANGE SCAN IDX-NOM
4 TABLE ACCESS BY ROWID COMMANDE
5 INDEX RANGE SCAN IDX-COMMANDE

Débat (5 Mn)

- Quels sont les points forts de l'approche RBO?
- Quels sont les points faibles ?

Optimisation des requêtes à base de règles

Limites

- La restructuration d'un arbre sous-entend souvent la permutation des opérateurs binaires
- Cette optimisation ignore la taille de chaque table, les facteurs de sélectivité, etc.
- Aucune estimation de résultats intermédiaires

Optimisation basée sur les modèles de coût
(Cost Based COB)

Optimisation des requêtes à base de coûts (CBO)

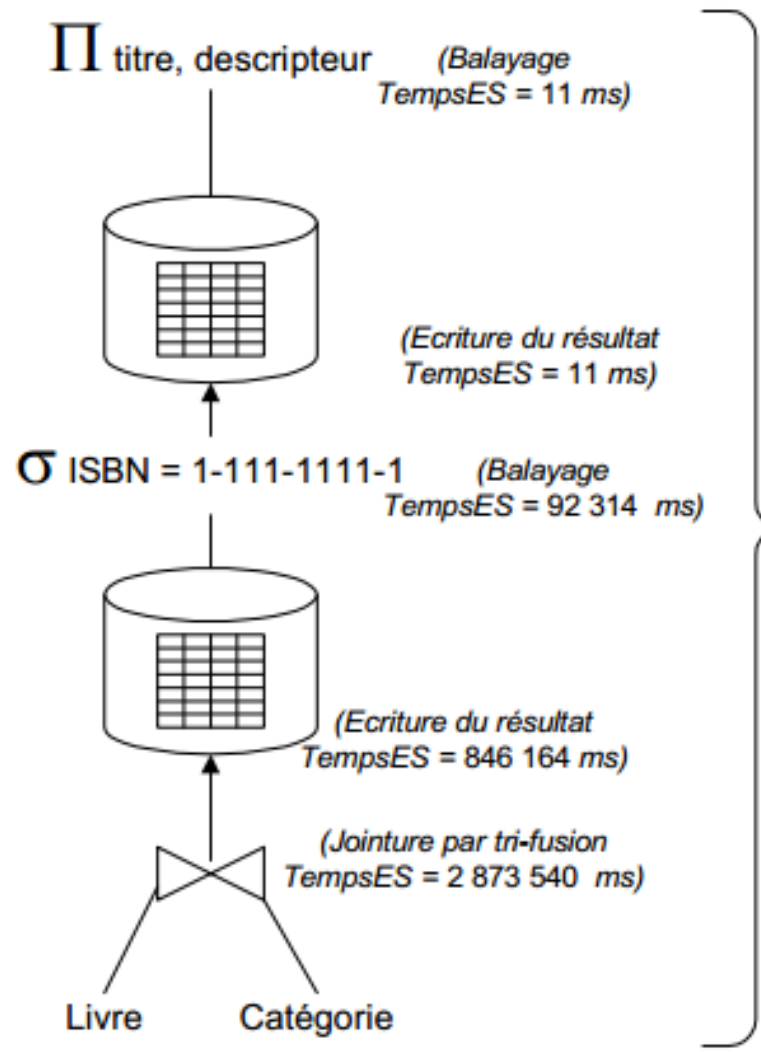
- Principe
- Sélectivité
  - Jointure
  - Projection
  - Séléction

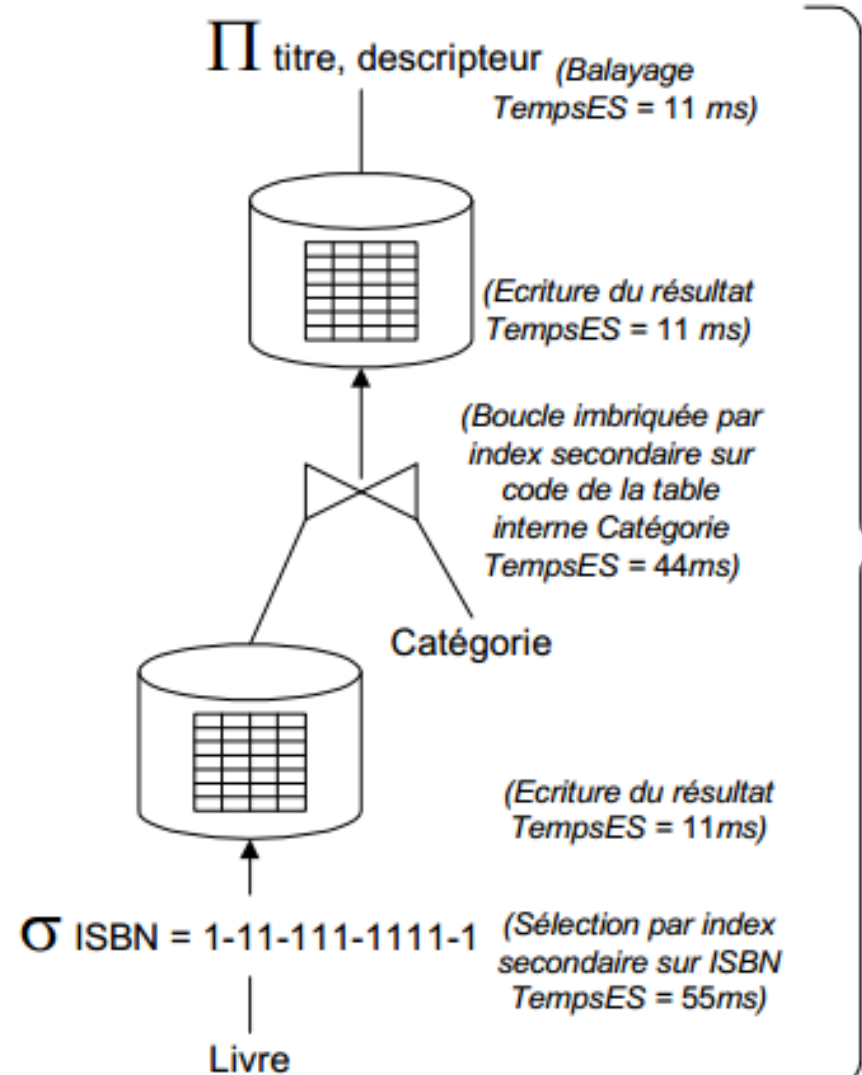
CBO: Méthodes d'optimisation basées sur les modèles de coûts

Optimisation des requêtes à base de coût

Motivation

```
SELECT titre, description ,
FROM Livre L, Catégorie C
WHERE L. id_Livre = C.id_Livre
AND ISBN= 1-111-1111-1
```





Optimisation des opérations: Notions (I)

- Les tables relationnelles sont stockées physiquement dans des fichiers sur disque

- Lorsqu 'on veut accéder aux données, on doit transférer le contenu pertinent des fichiers dans la mémoire

  - Fichier = séquence de tuples
  - Bloc (page) = unité de transfert de données entre disque et mémoire
  - Facteur de blocage = nombre de tuples d'une relation qui «tiennent» dans un bloc
  - Coût de lecture (ou écriture) d'une relation = nombre de blocs à lire du disque vers la mémoire (ou à écrire de la mémoire vers le disque)

Optimisation des requêtes à base de coût

Estimation de l'évaluation du coût d'un Plan

- Pour chaque plan, nous devrions être en mesure d'estimer le coût global.
- Pour chaque nœud de l'arbre de requête, nous estimons le coût d'exécution de l'opération correspondante.
- Pour chaque nœud de l'arbre de requête , nous estimer la taille du résultat, qui est utilisée par l'opération sur le nœud parent.
- Afin d'estimer correctement le coût de chaque opération et de la taille de son résultat, l'optimiseur utilise certaines informations statistiques gérées par le SGBD courant

Optimisation des requêtes à base de coût

Estimation de l'évaluation du coût d'un Plan

- Les coûts d'interrogation sont définies par le temps de réponse une requête (traiter le plan d'exécution de la requête)
- Différents facteurs contribuent aux coûts de la requête
  - –le temps d'accès au disque, le temps de CPU ou même le temps de communication de réseau
- Les coûts sont souvent dominées par le temps d'accès au disque
  - –temps de recherche (TS) (~ 4 ms)
  - –temps de transfert (TT) (par exemple 0,1 ms par bloc disque)
    - Les opérations d'écriture sont normalement plus lent que les opérations de lecture

Optimisation des requêtes à base de coût

Estimation de l'évaluation du coût d'un Plan

- Simplification
  - –nombre de blocs à transférer
- Jointures
  - –taille du résultat
- Contexte réparti
  - –seulement coût de transfert entre sites
- Hypothèses sur distributions

Optimisation basée sur modèle de coût

- Soit Q une requête à optimiser:
- Procédure:
  1. Énumérer tous les plans $\{P_1, \ldots, P_m\}$ pour chaque requête (notons que chaque requête possède un ensemble d'opérations $O_1, \ldots, O_k$)
  2. Pour chaque plan P_j

     Pour chaque opération O_i du plan P_j, énumérer les routines d'accès

     Sélectionner la routine ayant le coût le moins élevé

Coût(P_i) = $\Sigma_{(l=1, k)}$ min(O_l)

Coût (Q): $\Sigma_{(h=1, m)}$ Coût(P_h)

90

Optimisation des requêtes à base de coût

- Statistiques au sujet des tables

Statistique	Signification
NT	Nombre de lignes de la table T
TailleLigneT	La taille d'un ligne de la table T
FBT	Facteur de blocage moyen de T
FBMT	Facteur de blocage maximal de T Estimation : $\lfloor$(TailleBloc-TailleDescripteurBloc)/TailleLigneT$\rfloor$.
BT	Nombre de blocs de la table T Estimation : $\lceil$ NT / FBT $\rceil$

Statistique	Signification
$Card_T$ (Colonne):	Nombre de valeurs distinctes (cardinalité) de la colonne pour la table T
Min_T (Colonne):	Valeur minimum de la colonne dans la table T
Max_T (Colonne):	Valeur maximum de la colonne dans la table T

Optimisation des requêtes à base de coût

- Facteur de sélectivité
  - Proportion de tuples du produit cartésien des relations touchées qui satisfont une condition.

```
SELECT * FROM Customer WHERE age_Cust > 18;
```

FS = 1/NDIST(A) avec un modèle uniforme

```
SELECT * FROM Customer , Sale ;
```

FS= 1

Exemple : SF=?

```
SELECT *
FROM R1, R2
```

SF =1

```
SELECT *
FROM R1
WHERE A = valeur
```

SF = 1/NDIST(A) avec un modèle uniforme

Sélectivité des Restrictions

- Sélectivité des Restrictions
  - TAILLE (s(R)) = s * TAILLE(R) avec:

    s (A = valeur) = 1 / NDIST(A)

    s(A > valeur) = (max(A) - valeur) / (max(A) - min(A))

    s(A < valeur) = (valeur - min(A)) / (max(A) - min(A))

    s (A IN liste valeurs) = (1/NDIST(A)) * CARD(liste valeurs)

    s(P et Q) = s(P) * s(Q)

    s(P ou Q) = s(P) + s(Q) - s(P) * s(Q)

    s(not P) = 1 - s(P)

  - Le coût dépend de l'algorithme (index, hachage ou balayage).

Sélectivité des Projections

Sélectivité des Projections

- TAILLE(px(R)) = p(x) * (1-d) * TAILLE(R)
  - avec p(x) = Larg(x) / Larg(R)
  - d = probabilité de doubles
  - CARD(X) / CARD(DOM(X))**2

Sélectivité des Jointures

Sélectivité des Jointures

- TAILE(R1 ⋈ R2) = p * TAILLE(R1) * TAILLE(R2)
  - p dépend du type de jointure et de la corrélation des colonnes :
    - p = 0 si aucun tuple ne joint
    - p = 1 / MAX(NDIST(A),NDIST(B)) si distribution uniforme équiprobable des attributs A et B sur un même domaine
    - p = 1 si produit cartésien

- L'algorithme change radicalement les coûts
  - linéaire si index,
  - produit des tailles si boucles imbriquées.

Cardinalité des agrégations et des unions

Cardinalité de l'agrégation

Le nombre de tuples générés à partir d'une agrégation correspond au nombre des groupes résultants. Ce dernier peut être compris entre 1 et $|R|$, donc la cardinalité du résultat de l'agrégation est égale à :

$$|\gamma(R)| = \frac{|R|}{2}$$

Cardinalité de l'union

Le nombre maximal de tuples généré par l'union de deux relations R et S ($R \cup S$) est la somme de leurs cardinalités $|R|$ et $|S|$, et le minimum est le maximum entre $|R|$ et $|S|$. Ainsi, la cardinalité du l'union peut être la moyenne des valeurs maximales et minimales.

$$|R \cup S| = max(|R|, |S|) + \frac{min(|R|, |S|)}{2}$$

Cardinalité de l'intersection et de la différence

Cardinalité de l'intersection

La cardinalité du résultat de l'intersection de deux relations R et S ($R \cap S$) est comprise entre 0 et la cardinalité minimale des deux relations. Ainsi, la cardinalité peut être considérée comme la valeur moyenne :

$$|R \cap S| = \frac{min(|R|, |S|)}{2}$$

Cardinalité de la différence

Le maximum de tuples généré à partir de la différence de deux relations R et S ($R-S$), est la cardinalité de R ($|R|$) et le minimum est $|R| - |S|$. Ainsi, la cardinalité de la différence peut être considérée comme la moyenne :

$$|R - S| = \frac{(|R| - |S|)}{2}$$

Estimation de la sélectivité des prédicats complexes

Prédicat p	Sélectivité f_p
$\neg(p)$	$1 - f_p$
$p_1 \wedge p_2$	$f_{p_1} \times f_{p_2}$
$p_1 \vee p_2$	$f_{p_1} + f_{p_2} - f_{p_1} \times f_{p_2}$
$a = val$	$\frac{1}{\lvert a \rvert}$
$a = b$	$\frac{1}{max(\lvert a \rvert, \lvert b \rvert)}$
$a > val$	$\frac{(max_a - val)}{(max_a - min_a)}$
$a < val$	$\frac{(val - min_a)}{(max_a - min_a)}$
$val_1 \leqslant a \leqslant val_2$	$\frac{(val_2 - val_1)}{(max_a - min_a)}$

Tableau A.1 – Notation pour l'estimation de cardinalité.

Paramètres	Description
R, S	Deux relations ou résultat intermédiaire
$\lvert R \rvert$	Nombre de tuples de la relation R
$\lvert\lvert R \rvert\rvert$	Nombre de pages sur lesquelles la relation R est stockée
$\lvert a \rvert$	Nombre de valeurs distinctes de l'attribut a
max_a	Valeur maximale pour un attribut a dans la relation R
min_a	Valeur minimale pour un attribut a dans la relation R
$L_a(a_R)$	Longueur moyenne de la valeur de l'attribut a de la relation R (en octets)
$L_t(R)$	Longueur moyenne d'un tuple de relation R (en octets)