

SAM 4I803

Cours 3 : Traitement et optimisation de requêtes

diapos à connaître par cœur 

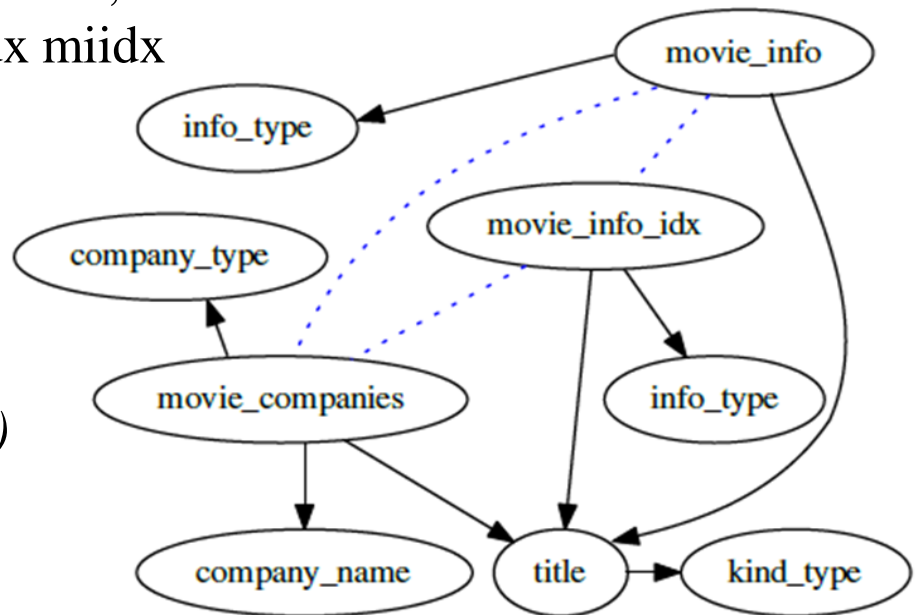


Intro

- La base de films IMDB
 - <https://www.imdb.com/interfaces/>
- Benchmark réaliste
 - Ref: How Good Are Query Optimizers, Really?
 - VLDB 2015 <http://www.vldb.org/pvldb/vol9/p204-leis.pdf>
 - Cast_info: 36M tuples
 - Movie_info: 15M tuples
 - Données du benchmark
 - <ftp://ftp.fu-berlin.de/pub/misc/movies/database/>

Requête réelle

```
SELECT cn.name, mi.info, miidx.info
FROM company_name cn, company_type ct,
      info_type it, info_type it2, title t,
      kind_type kt, movie_companies mc,
      movie_info mi, movie_info_idx miidx
WHERE cn.country_code = '[us]'
AND ct.kind = 'production companies'
AND it.info = 'rating'
AND it2.info = 'release dates'
AND kt.kind = 'movie'
AND prédicats de jointure (11 égalités)
```



Toutes les requêtes: <http://db.in.tum.de/people/sites/leis/qo/job.tgz>



Questions

- Cardinalité
 - Estimations précises → requête rapide ?
 - Estimations très approximatives → requêtes lentes ?
- Ordre des jointures important ?
- Important d'explorer beaucoup de plans ?

Problème

EMP(ENO, ENAME, TITLE)

PROJECT(PNO, PNAME, BUDGET)

WORKS(ENO, PNO, RESP, DUR)

Soit la requête

pour chaque projet de budget > 250 qui emploie plus de 2 employés, donner le nom et le titre des employés

Comment l'exprimer en SQL ?

Un plan d'exécution possible

```
SELECT DISTINCT Ename, Title
FROM Emp, Project, Works
WHERE Budget > 250
AND Emp.Eno=Works.Eno
AND Project.Pno=Works.Pno
AND Project.Pno IN
  (SELECT Pno
   FROM Works
   GROUP BY Pno
   HAVING COUNT(*) > 2)
```

$T_1 \leftarrow$ Lire la table Project et sélectionner
les tuples de Budget > 250

$T_2 \leftarrow$ Joindre T_1 avec la relation Works

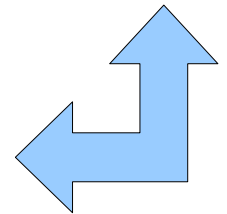
$T_3 \leftarrow$ Joindre T_2 avec la relation Emp

$T_4 \leftarrow$ Grouper les tuples de Works sur Pno et pour les groupes qui
ont plus de 2 tuples, projeter sur Pno

$T_5 \leftarrow$ Joindre T_3 avec T_4

$T_6 \leftarrow$ Projeter T_5 sur Ename, Title

Résultat \leftarrow Éliminer doublons dans T_6



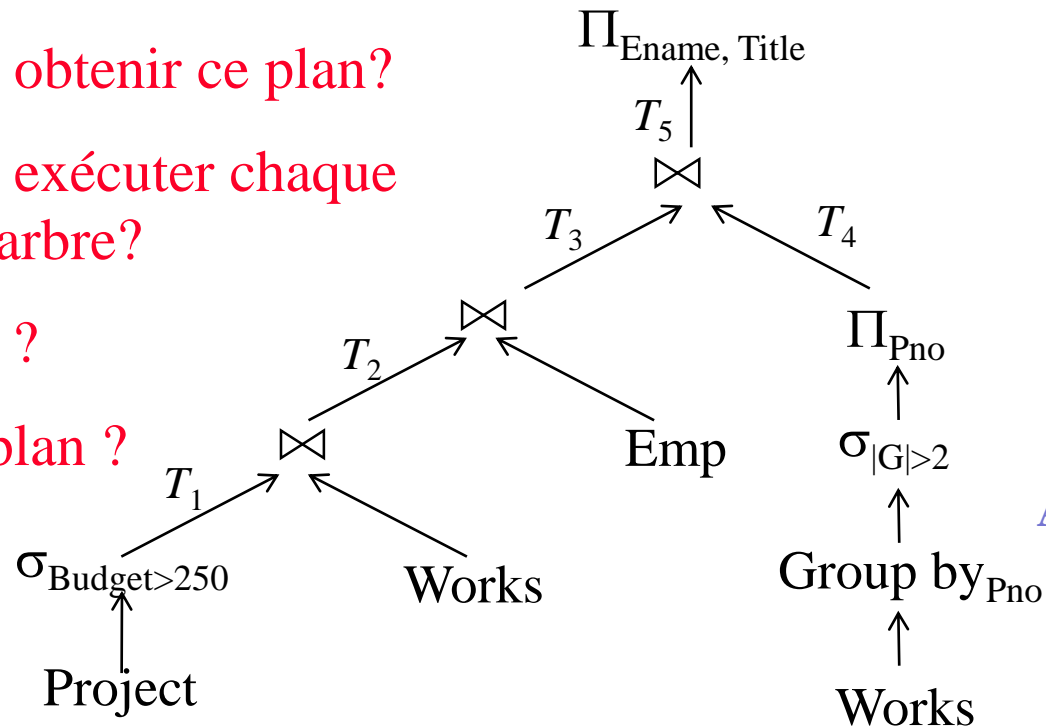
Représentation algébrique

1. Comment obtenir ce plan?

2. Comment exécuter chaque nœud/sous-arbre?

3. Quel coût ?

4. Meilleur plan ?

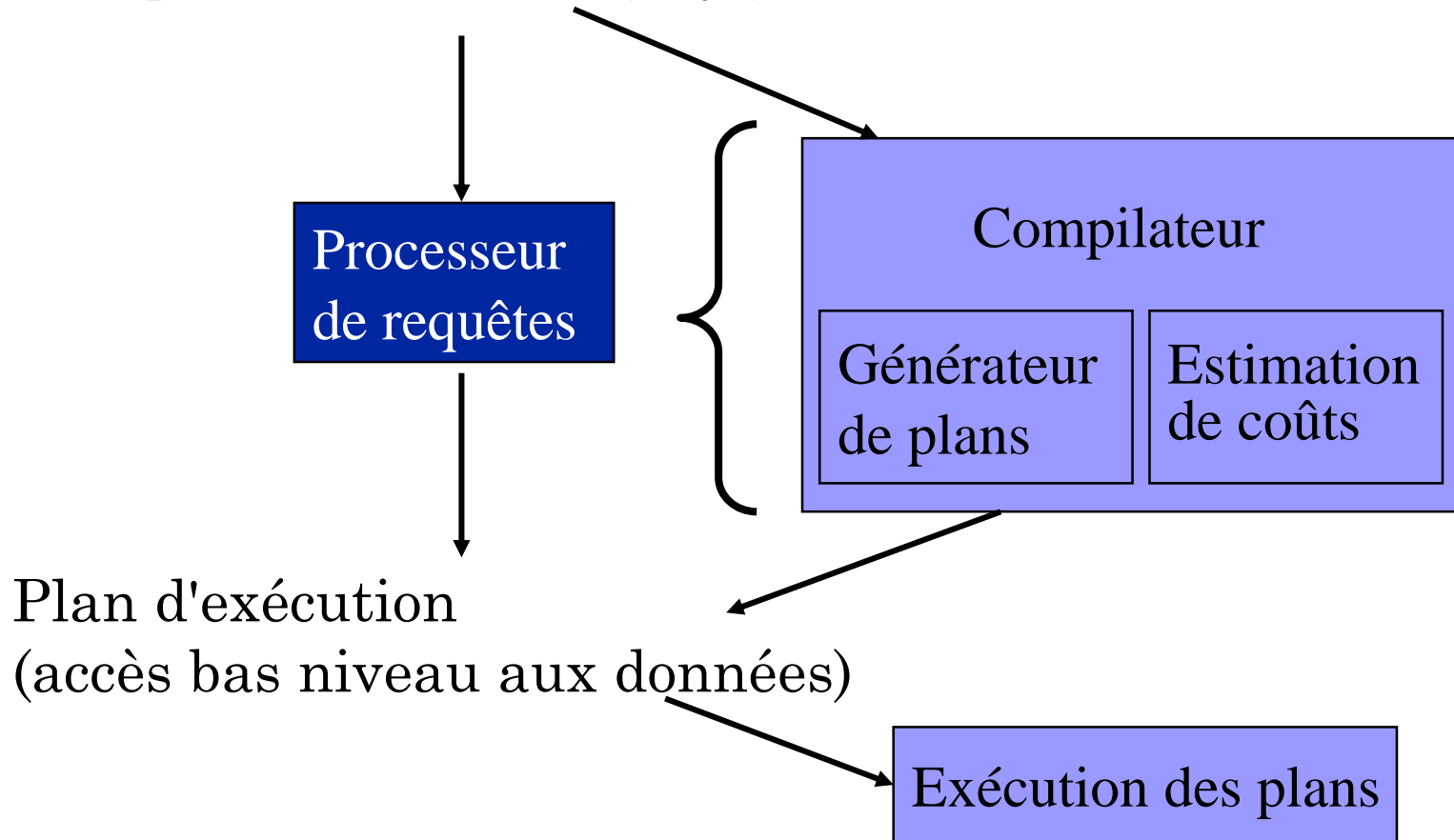


Algèbre étendue

$\Pi_{\text{Ename, Title}}(\Pi_{\text{Pno}}(\sigma_{|G|>2} \text{Group}_{\text{Pno}}(\text{Works})) \bowtie$
 $(\text{Emp} \bowtie ((\sigma_{\text{Budget}>250000} \text{Project}) \bowtie \text{Works})))$

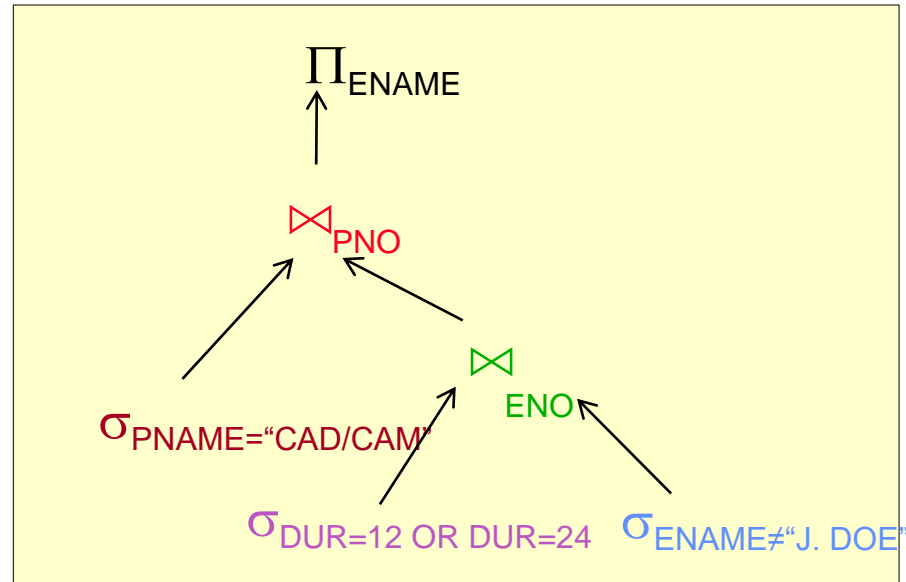
• Traitement des requêtes

Requête déclarative (SQL)

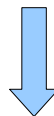


Exécution d'un plan

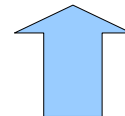
MÉMOIRE
CENTRALE



Écriture données temp.

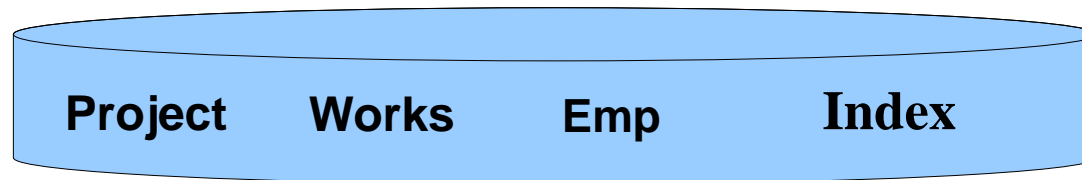


pages

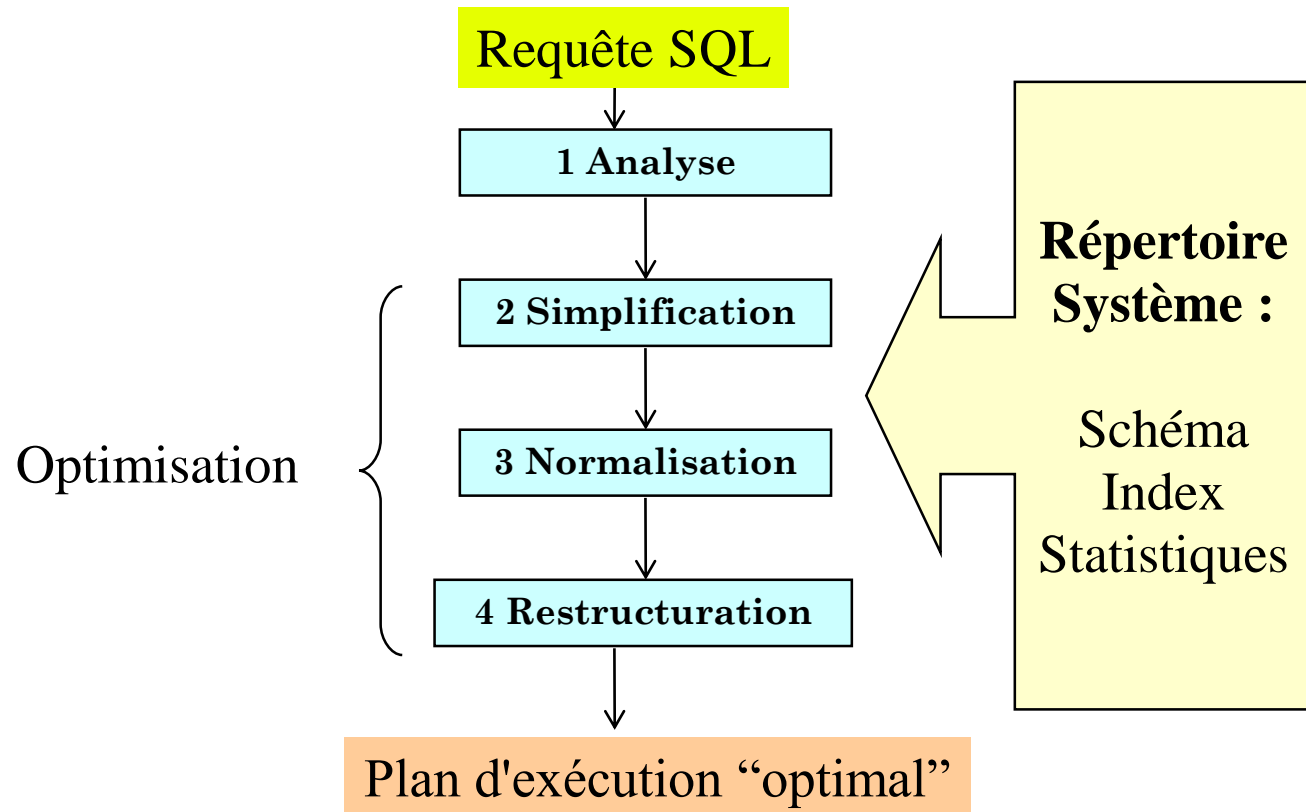


Lecture données / index

DISQUE



Étapes du traitement d'une requête



Normalisation de requête

- Analyse lexicale et syntaxique
 - vérification de la validité de la requête
 - vérification des attributs et relations
 - vérification du typage de la qualification
- Mise de la requête en **forme normale**
 - forme normale conjonctive
$$(p_{11} \vee p_{12} \vee \dots \vee p_{1n}) \wedge \dots \wedge (p_{m1} \vee p_{m2} \vee \dots \vee p_{mn})$$
 - forme normale disjonctive
$$(p_{11} \wedge p_{12} \wedge \dots \wedge p_{1n}) \vee \dots \vee (p_{m1} \wedge p_{m2} \wedge \dots \wedge p_{mn})$$
 - OR devient union
 - AND devient jointure ou sélection

Simplification

- Pourquoi simplifier?
 - plus une requête est simple, plus son exécution peut être efficace
- Comment? en appliquant des transformations
 - élimination de la redondance
 - règles d'idempotence
$$p_1 \wedge \neg(p_1) \equiv \text{faux}$$
$$p_1 \wedge (p_1 \vee p_2) \equiv p_1$$
$$p_1 \vee \text{faux} \equiv p_1$$
$$\dots$$
 - application de la transitivité (att1=att2 ,att2=att3)
- Éliminer des opérations redondantes : **élagage**
 - ex. : pas besoin de distinct après une projection sur une clé
- utilisation des règles d'intégrité
 - CI : att1 < 100 Q: ... where att1 > 1000...

Exemple de simplification

```
SELECT      Title
FROM        Emp
WHERE       Ename = 'J. Doe'      P1
OR          (NOT (Title = 'Programmer')  -P2
AND         (Title = 'Programmer'      P2
OR          Title = 'Elect. Eng.')    P3
AND         NOT (Title = 'Elect. Eng.')) -P3
```



$P1 \vee (\neg P2 \wedge (P2 \vee P3) \wedge \neg P3)$

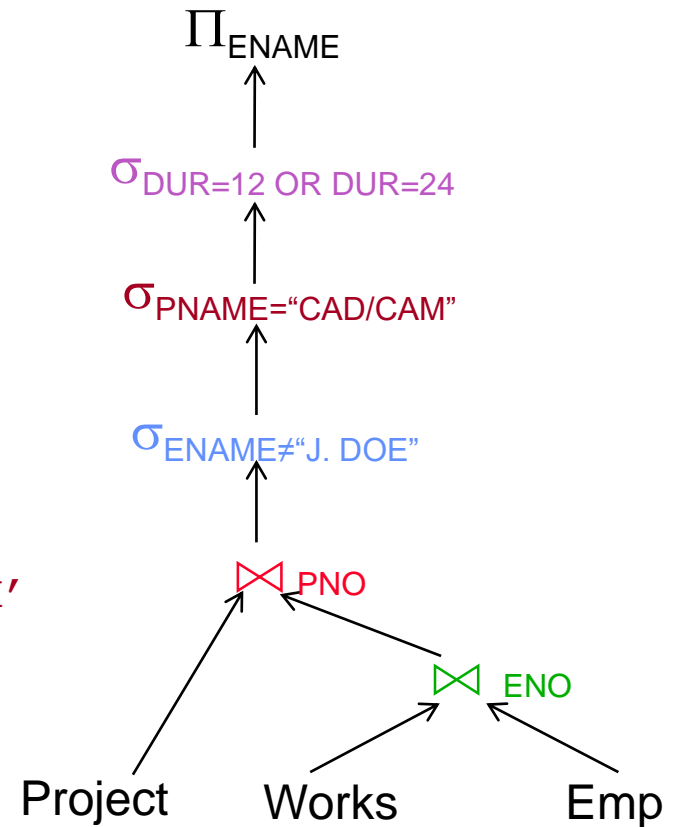
```
SELECT      Title
FROM        Emp
WHERE       Ename = 'J. Doe'
```

Traduction en algèbre

Conversion en arbre algébrique

Exemple :

```
SELECT   Ename
FROM     Emp, Works, Project
WHERE    Emp.Eno = Works.Eno
AND      Works.Pno = Project.Pno
AND      Emp.Ename <> 'J.Doe'
AND      Project.name = 'CAD/CAM'
AND      (Works.Dur=12 OR
           Works.Dur=24)
```



Heuristiques

Observation : les opérations manipulant moins de données sont plus rapides (sélectivité corrélée à la performance)

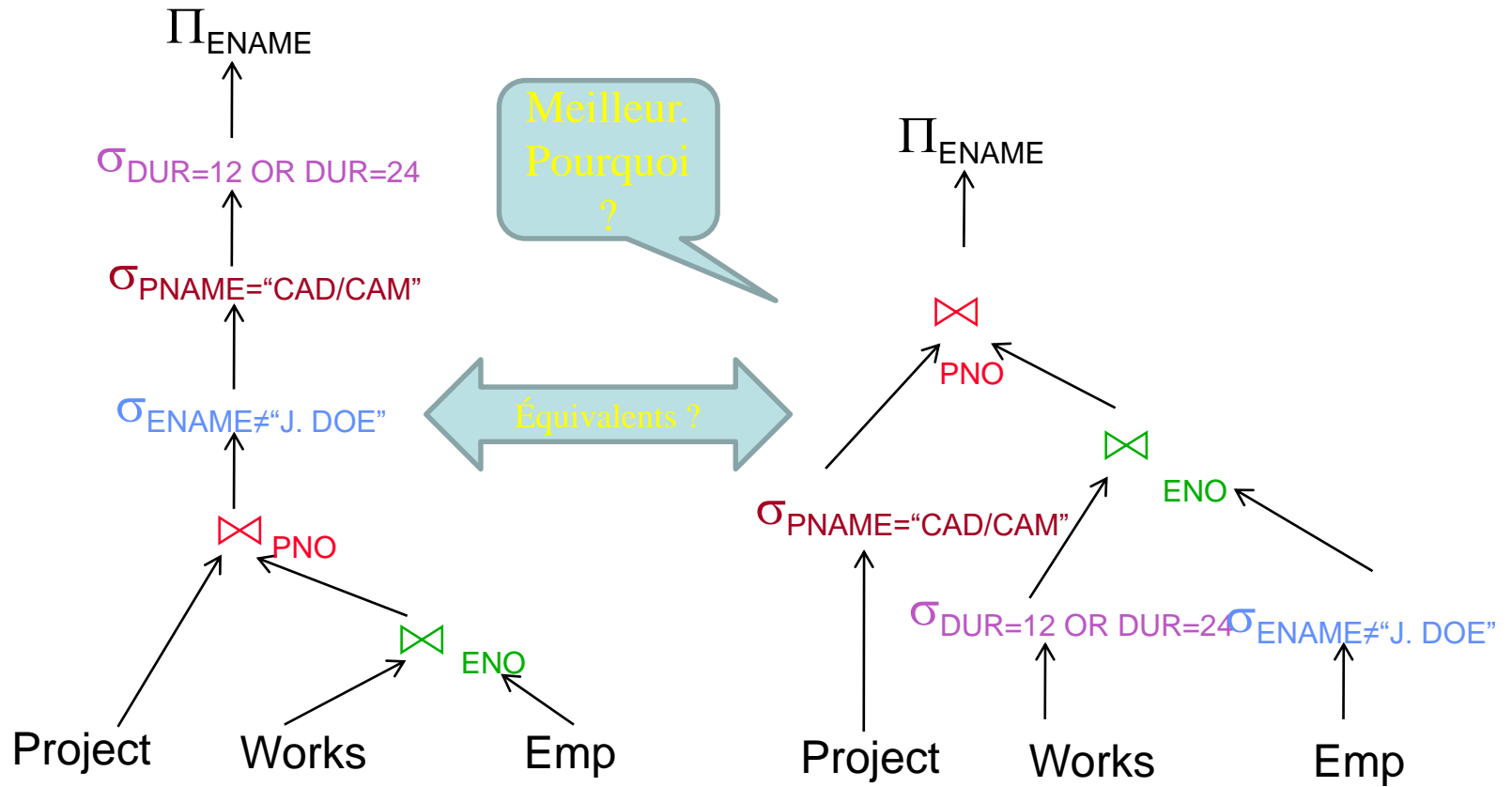
Objectif : déterminer un ordre pour les opérations, censé être efficace.

Méthode:

Commencer par traiter les opérations les **plus sélectives** (projection, sélection) et de manière à réduire la taille des données d'entrée pour les opérateurs suivants (jointures).

La place en mémoire est un facteur primordial pour l'efficacité d'une jointure (cf. algo de jointure, cours suivant)

Exemple



Optimisation basée sur le coût

- Elaborer des plans
 - arbre algébrique, restructuration, ordre d'évaluation
- Estimer leurs coûts
 - fonctions de coût
 - en terme de temps d'exécution
 - coût I/O + coût CPU
 - poids très différents
 - par ex. coût I/O = 1000 * coût CPU
- Choisir le meilleur plan
 - Espace de recherche : ensemble des expressions algébriques équivalentes pour une même requête
 - algorithmes de recherche:
 - parcourir l'espace de recherche
 - algorithmes d'optimisation combinatoire

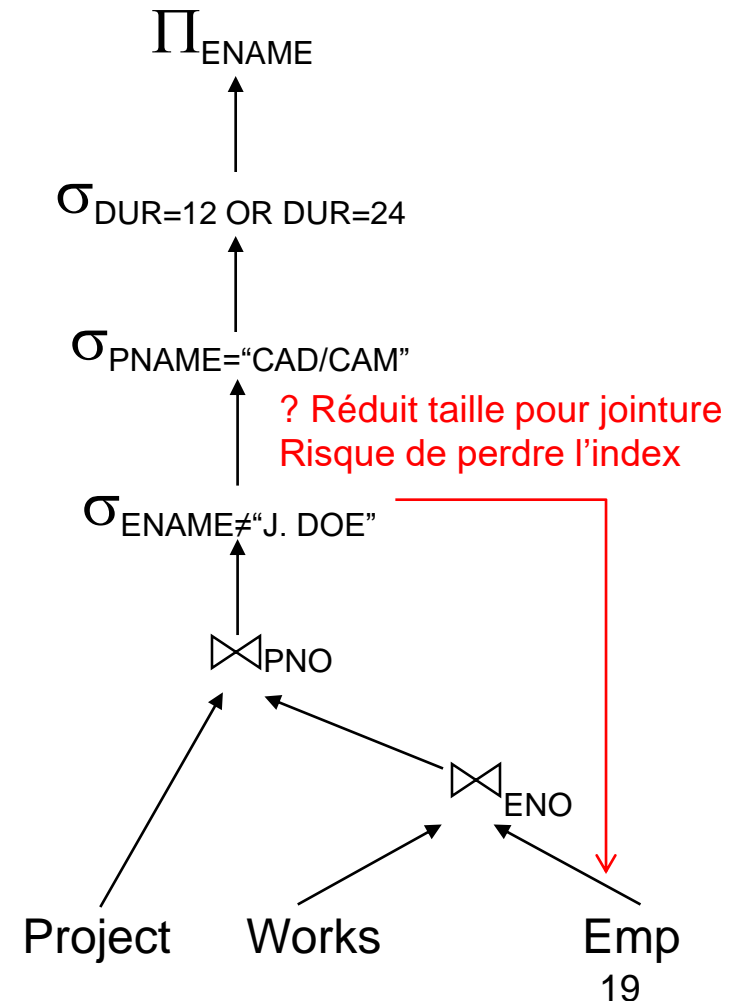
Restructuration

- Objectif : choisir l'ordre d'exécution des opérations algébriques (élaboration du plan logique).
- Conversion en arbre algébrique
- Transformation de l'arbre (optimisation)
 - règles de transformation (équivalence algébriques),
 - estimation du coût des opérations en fonction de la taille
 - Estimation du résultat intermédiaire (taille et ordre?)
 - En déduire l'ordre des jointures

Restructuration

- Conversion en arbre algébrique
- Exemple

```
SELECT  Ename
FROM    Emp, Works, Project
WHERE   Emp.Eno=Works.Eno
AND     Works.Pno=Project.Pno
AND     Ename NOT='J.Doe'
AND     Pname = 'CAD/CAM'
AND     (Dur=12 OR Dur=24)
```



Coût d'un plan

- Fonction de coût = estimation du temps écoulé pour
 - Accéder aux données :
 - temps I/O exprimé en nombre de pages lues ou écrites
 - Calculer le résultat d'une opération à partir des données lues
 - Temps CPU dépend du nombre d'instructions
- Estimation du coût d'exécution de chaque noeud de l'arbre algébrique
 - utilisation de pipelines ou de relations temporaires importante
 - Pipeline : les tuples sont passés directement à l'opérateur suivant.
 - Pas de relations intermédiaires (petites mémoires, ex. carte à puce).
 - Permet de paralléliser (BD réparties, parallèle)
 - Intéressant même pour cas simples : $\sigma_{F \wedge F'}(R)$, index sur $F' \rightarrow \sigma_F(\sigma_{F'}(R))$
 - Relation temporaire : permet de trier mais coût de l'écriture

Estimer la **cardinalité** d'une opération

- Estimation du **nombre de nuplets** résultant de chaque nœud par rapport à ses entrées
 - Permet d'estimer le coût de l'opération suivante
 - sélectivité des opérations – “facteur de réduction”
 - propagation d'erreur possible
 - basé sur les statistiques maintenues par le SGBD

Statistiques

- Relation
 - cardinalité : $\text{card}(\mathbf{R})$
 - taille d'un tuple : $\text{largeur}(\mathbf{R})$
 - fraction de tuples participant une jointure / attribut
 - ...
- Attribut
 - cardinalité du domaine
 - nombre de valeurs distinctes $\mathbf{D}(\mathbf{R}, \mathbf{A}) = \Pi_{\mathbf{A}}(\mathbf{R})$
 - Valeur max, valeur min
- Hypothèses
 - Indépendance entre différentes valeurs d'attributs
 - Distribution uniforme des valeurs d'attribut dans leur domaine
 - Sinon, il faut maintenir des histogrammes (voir diapo suivante)
- Stockage :
 - Les statistiques sont des métadonnées, stockées sous forme relationnelle (cf. TME)
 - Rafraîchies périodiquement, pas à chaque fois.

Le fameux
compromis L/E

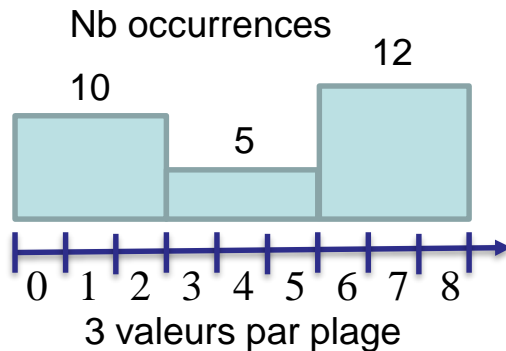
Histogrammes

Sert à estimer la cardinalité de la requête :
Select * from R where A = 8 ?



Histogramme Equilarge

Plages de valeurs de même taille.
Hypothèse d'uniformité
dans un intervalle

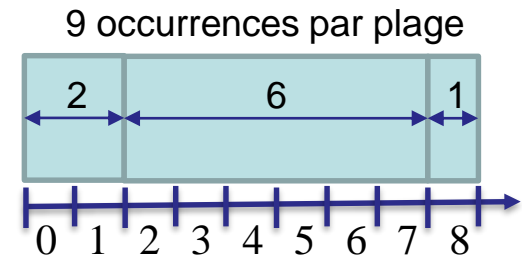


$$\text{card} = 12 / 3 = 4$$



Histogramme Equiprofond

Plages de valeurs contenant le même
nombre d'occurrences.
Plus précis pour les valeurs fréquentes.



$$\text{Card} = 9$$

A	card
0	7
1	2
2	1
3	5
4	0
5	0
6	2
7	1
8	9

Card **estimée** pour A=8 ?

Cardinalité réelle = **9**



Cardinalité d'une sélection

Facteur de sélectivité

- Soit la **sélection** notée : $\sigma_{pred}(R)$
 - $pred$ est le prédicat de sélection
 - En SQL : `select * from R where pred`

- **Cardinalité** d'une sélection

$$card(\sigma_{pred}(R)) = SF(pred) * card(R)$$

SF est une *estimation* de la **sélectivité du prédicat** $pred$, dont la forme générale est "nombre d'éléments sélectionnés / nombre d'éléments possibles"



Facteur de sélectivité :

Sélection sur **un** attribut

- **Egalité** : sélection de valeurs
 - Nbval: nombre de valeurs sélectionnées
 - **$SF = Nbval / D(R,A)$**
- **Inégalité** : sélection d'un intervalle
 - Intervalle = segment d'un domaine (continu)
 - Bornes du domaine : $A \in [\min(A), \max(A)]$
 - Longueur du domaine : $L(A) = \max(A) - \min(A)$
 - **$SF = \text{longueur du segment sélectionné} / L(A)$**

pred	Nbval
$A = v$	1
$A \text{ in } (v1, v2, \dots, vk)$	k
$A = v1 \text{ OR } A = v2$ $\text{OR } \dots \text{ OR } A = vk$	k

pred	Longueur
$A < v$	$v - \min(A)$
$A > v$	$\max(A) - v$
$A \geq v1 \text{ and } A \leq v2$ s'écrit aussi: $A \text{ between } v1 \text{ and } v2$	$v2 - v1$
$A \leq v1 \text{ or } A \geq v2$	$\max(A) - v2 + v1 - \min(A)$



Facteur de sélectivité

Sélection sur **plusieurs** attributs

- Le prédicat est **composé** de plusieurs prédicats:
 - $\text{pred}_A, \text{pred}_B, \dots$
 - Exple: $\text{pred}_{\text{age}}: \text{age} > 20$ $\text{pred}_{\text{ville}}: \text{ville} = \text{'Paris'}$
- Conjonction: pred_A **AND** pred_B
 - $\text{SF}(\text{pred}_A \text{ AND } \text{pred}_B) = \text{SF}(\text{pred}_A) * \text{SF}(\text{pred}_B)$
 - Exple:
 - Pred : $\text{age} > 20 \text{ AND } \text{ville} = \text{'Paris'}$
 - $\text{SF}(\text{pred}) = \text{SF}(\text{age} > 20) * \text{SF}(\text{ville} = \text{'Paris'})$
- Disjonction entre 2 termes : pred_A **OR** pred_B
 - $\text{SF}(\text{pred}_A \text{ OR } \text{pred}_B) = \text{SF}(\text{pred}_A) + \text{SF}(\text{pred}_B) - \text{SF}(\text{pred}_A) * \text{SF}(\text{pred}_B)$
 - Exple:
 - Pred : $\text{age} > 20 \text{ OR } \text{ville} = \text{'Paris'}$
 - $\text{SF}(\text{pred}) = \text{SF}(\text{age} > 20) + \text{SF}(\text{ville} = \text{'Paris'}) - \text{SF}(\text{age} > 20) * \text{SF}(\text{ville} = \text{'Paris'})$
- Rmq: peut se généraliser pour n prédicats
 - Disjonction n-aire: $(n-1)$ disjonctions binaires

Facteur de sélectivité

Sélection avec **négation**

- Négation: complément à 1
- $SF(\text{not}(\text{pred})) = 1 - SF(\text{pred})$
- Exemple:
 - `jourFermeture = 'Dimanche'`
 - $SF(\text{jourFermeture} = \text{'Dimanche'}) = 1/7$
 - `jourFermeture <> 'Dimanche'`
 - $SF(\text{jourFermeture} <> \text{'Dimanche'}) = 1 - 1/7 = 6/7$

Cardinalité des opérations

Projection

$$\text{card}(\Pi_A(R)) \leq \text{card}(R) \quad (\text{égalité si } A \text{ est } \textit{unique})$$

Produit cartésien

$$\text{card}(R \times S) = \text{card}(R) \cdot \text{card}(S)$$

Union

$$\text{borne sup. : } \text{card}(R \cup S) = \text{card}(R) + \text{card}(S)$$

$$\text{borne inf. : } \text{card}(R \cup S) = \max\{\text{card}(R), \text{card}(S)\}$$

Différence

$$\text{borne sup. : } \text{card}(R - S) = \text{card}(R)$$

$$\text{borne inf. : } 0$$



Cardinalité d'une jointure naturelle

- Jointure naturelle entre 2 tables R et S sur l'attribut A
 - A est clé "primary key" de R(A, ...)
 - donc $\text{card}(R) / D(R, A) = 1$
 - A est clé étrangère "foreign key" dans S (..., A*, ...)
 - $\text{domaine}(S.A) \subseteq \text{domaine}(R.A)$
 - Pour chaque tuple de S, il existe **un et un seul** tuple de R
- $\text{card}(R \bowtie_A S) = \text{card}(S) \cdot 1$
- Exple :
 - **Etu**(nomE, age) **Note**(nomE*, codeUE*, note) **UE**(codeUE, niveau, titre)
 - 100 Etudiants, 600 notes, 30 UE
 - $\text{card}(\text{Etu} \bowtie_{\text{nomE}} \text{Note}) = \text{card}(\text{Note}) = 600$
 - $\text{card}(\text{Note} \bowtie_{\text{codeE}} \text{UE}) = \text{card}(\text{Note}) = 600$
 - $\text{card}(\text{Etu} \bowtie_{\text{nomE}} \text{Note} \bowtie_{\text{codeE}} \text{UE}) = \text{card}(\text{Note}) = 600$



Cardinalité d'une jointure entre deux clés étrangères

- Cas d'une jointure $R \bowtie_A S$ entre 2 clés étrangères
 - **Entité**(A, ...), $R(..., A^*, ...)$, $S(..., A^*, ...)$
 - $\text{dom}(R.A) \subseteq \text{dom}(\text{Entité}.A)$ donc $D(\text{Entité}, A) \geq D(R, A)$
 - $\text{dom}(S.A) \subseteq \text{dom}(\text{Entité}.A)$ donc $D(\text{Entité}, A) \geq D(S, A)$
- $\text{card}(R \bowtie_A S) = \text{card}(R) * \text{card}(S) / D(\text{Entité}, A)$
- Exple :
 - **Etu**(nomE, age) et $D(\text{Etu}, \text{nomE}) = 100$
 - **InscritSport**(sport, nomE*, annee)
 - $D(\text{InscritSport}, \text{nomE}) = 20$
 - **Résa** (codeLivre*, nomE*, date, bibliothèque, durée)
 - $D(\text{Resa}, \text{NumE}) = 50$
 - 100 Etudiants, 40 InscritSport, 200 Résa de livres
 - $\text{card}(\text{InscritSport} \bowtie_{\text{nomE}} \text{Résa}) = 40 * 200 / 100 = 80$



Cardinalité d'une jointure entre 2 sélections

- Jointure entre deux expressions contenant des sélections
- Réécrire la jointure :
 - Exprimer la jointure avant la sélection
 - Commutativité de la composition des opérations :
 - $\text{jointure}(\text{sélection}, \text{sélection}) = \text{sélection}(\text{sélection}(\text{jointure}))$

$$\text{card}(\sigma_{p1}(R) \bowtie_A \sigma_{p2}(S)) =$$

$$\text{SF}(p1) * \text{SF}(p2) * \text{card}(R \bowtie_A S)$$

Espace de recherche

- Caractérisé par les plans “équivalents” pour une même requête
 - ceux qui donnent le même résultat
 - générés en appliquant les règles de transformation vues précédemment
- Le coût de chaque plan est en général différent
- L'ordre des jointures est important

Règles de transformation

- Commutativité des opérations binaires

- $R \times S \equiv S \times R$

- $R \bowtie S \equiv S \bowtie R$

- $R \cup S \equiv S \cup R$

- Associativité des opérations binaires

- $(R \times S) \times T \equiv R \times (S \times T)$

- $(R \bowtie S) \bowtie T \equiv R \bowtie (S \bowtie T)$

- Idempotence des opérations unaires

- $\Pi_A(\Pi_A(R)) \equiv \Pi_A(R)$

- $\sigma_{p_1(A_1)}(\sigma_{p_2(A_2)}(R)) \equiv \sigma_{p_1(A_1) \wedge p_2(A_2)}(R)$

- où $R[A]$ et $A' \subseteq A, A'' \subseteq A$ et $A' \subseteq A''$

Règles de transformation

- Commutativité de la sélection et de la projection (si proj. des attr. sél.)
- Commutativité de la sélection avec les opérations binaires

$$\sigma_{p(A)} (R \times S) \equiv (\sigma_{p(A)} (R)) \times S$$

$$\sigma_{p(A_i)} (R \bowtie_{(A_j B_k)} S) \equiv (\sigma_{p(A_i)} (R)) \bowtie_{(A_j B_k)} S$$

$$\sigma_{p(A_i)} (R \cup T) \equiv \sigma_{p(A_i)} (R) \cup \sigma_{p(A_i)} (T)$$

où A_i appartient à R et T

- Commutativité de la projection avec les opérations binaires

$$\Pi_C (R \times S) \equiv \Pi_{A'} (R) \times \Pi_{B'} (S)$$

$$\Pi_C (R \bowtie_{(A_j B_k)} S) \equiv \Pi_{A'} (R) \bowtie_{(A_j B_k)} \Pi_{B'} (S)$$

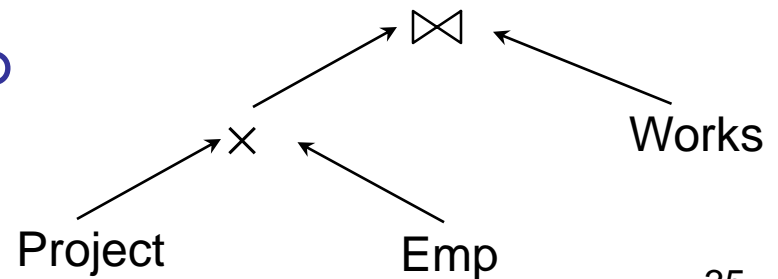
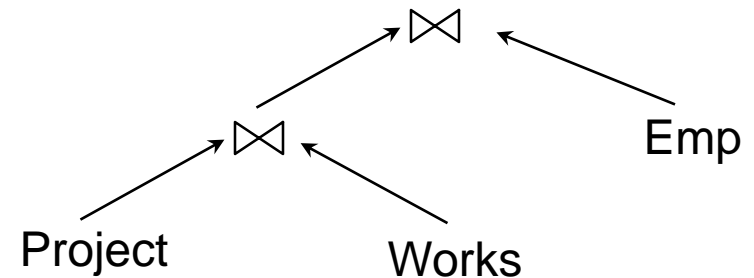
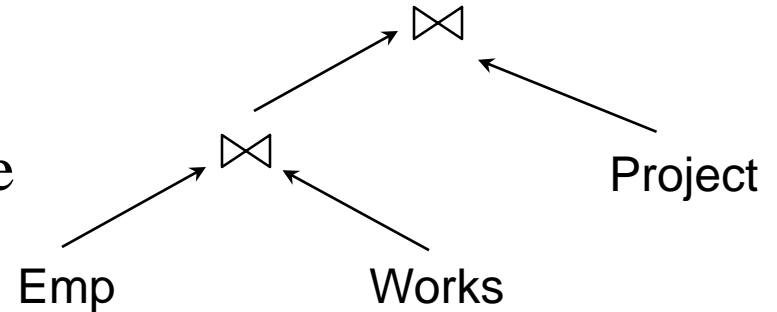
$$\Pi_C (R \cup S) \equiv \Pi_C (R) \cup \Pi_C (S)$$

où $R[A]$ et $S[B]$; $C = A' \cup B'$ où $A' \subseteq A$, $B' \subseteq B$, $A_j \subseteq A'$, $B_k \subseteq B'$

Ordre des jointures

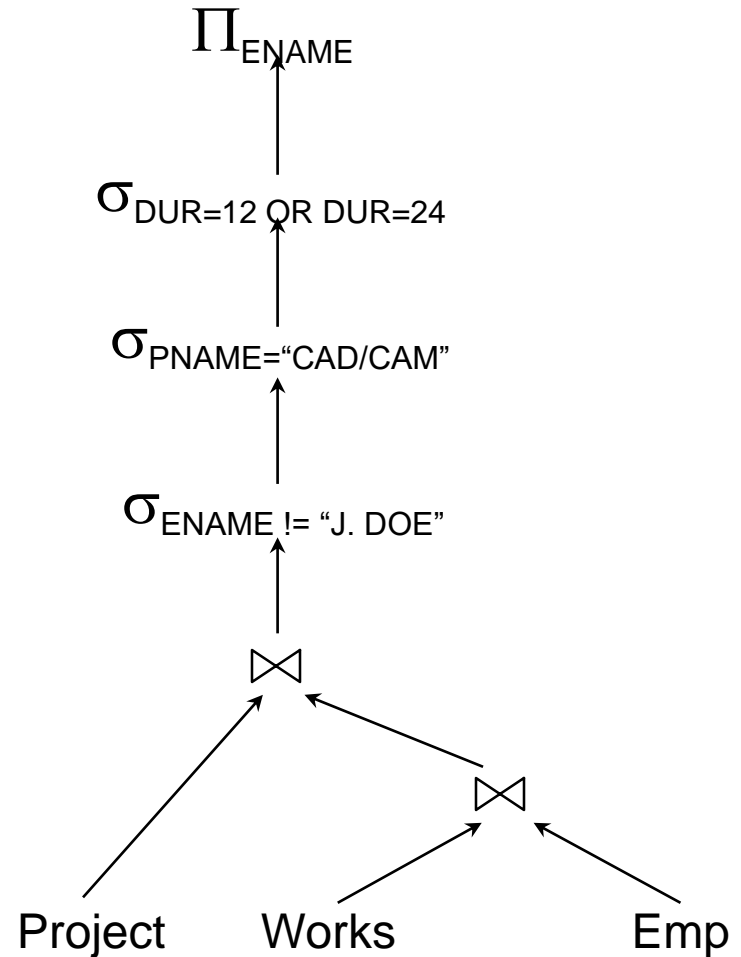
- Avec N relations, il y a $O(N!)$ ordres de jointures équivalents qui peuvent être obtenus en appliquant les règles de *commutativité* et d'*associativité*

```
SELECT  Ename, Resp
FROM    Emp, Works, Project
WHERE   Emp.Eno=Works.Eno
AND     Works.PNO=Project.PNO
```

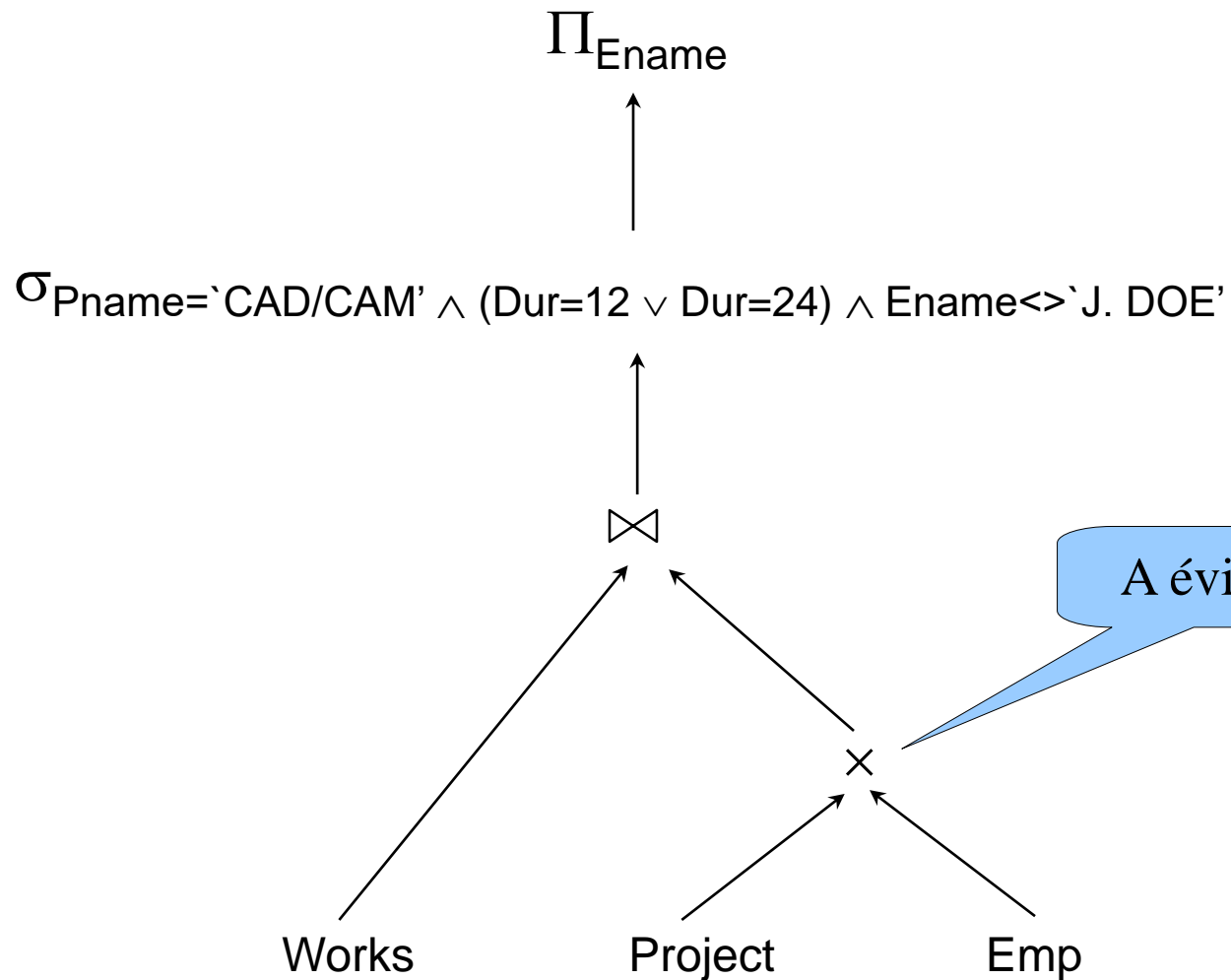


Exemple

```
SELECT Ename
FROM Project p, Works w,
      Emp e
WHERE w.Eno=e.Eno
AND   w.Pno=p.Pno
AND   Ename<>`J. Doe`
AND   p.Pname=`CAD/CAM`
AND   (Dur=12 OR Dur=24)
```

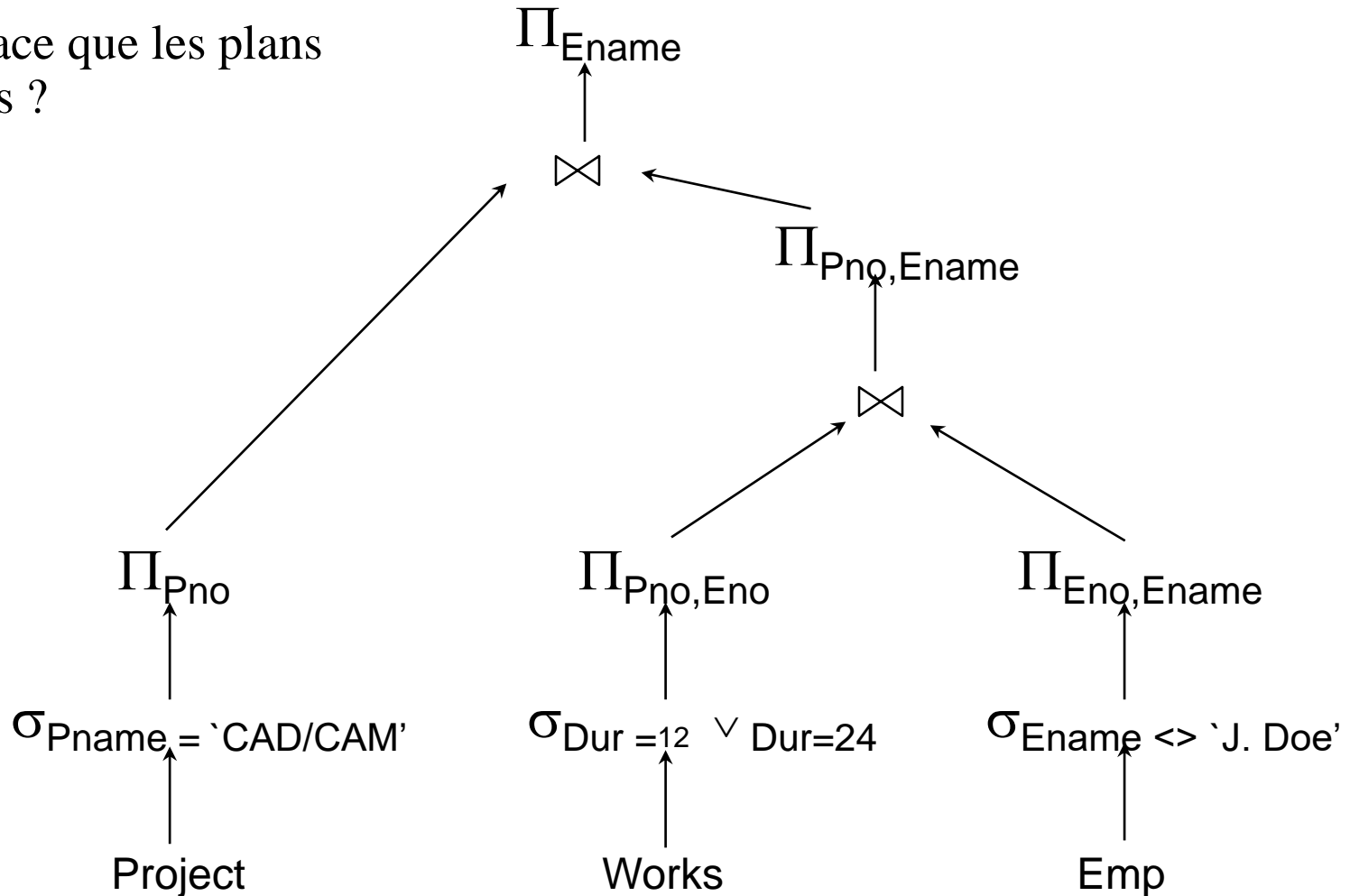


Plan équivalent

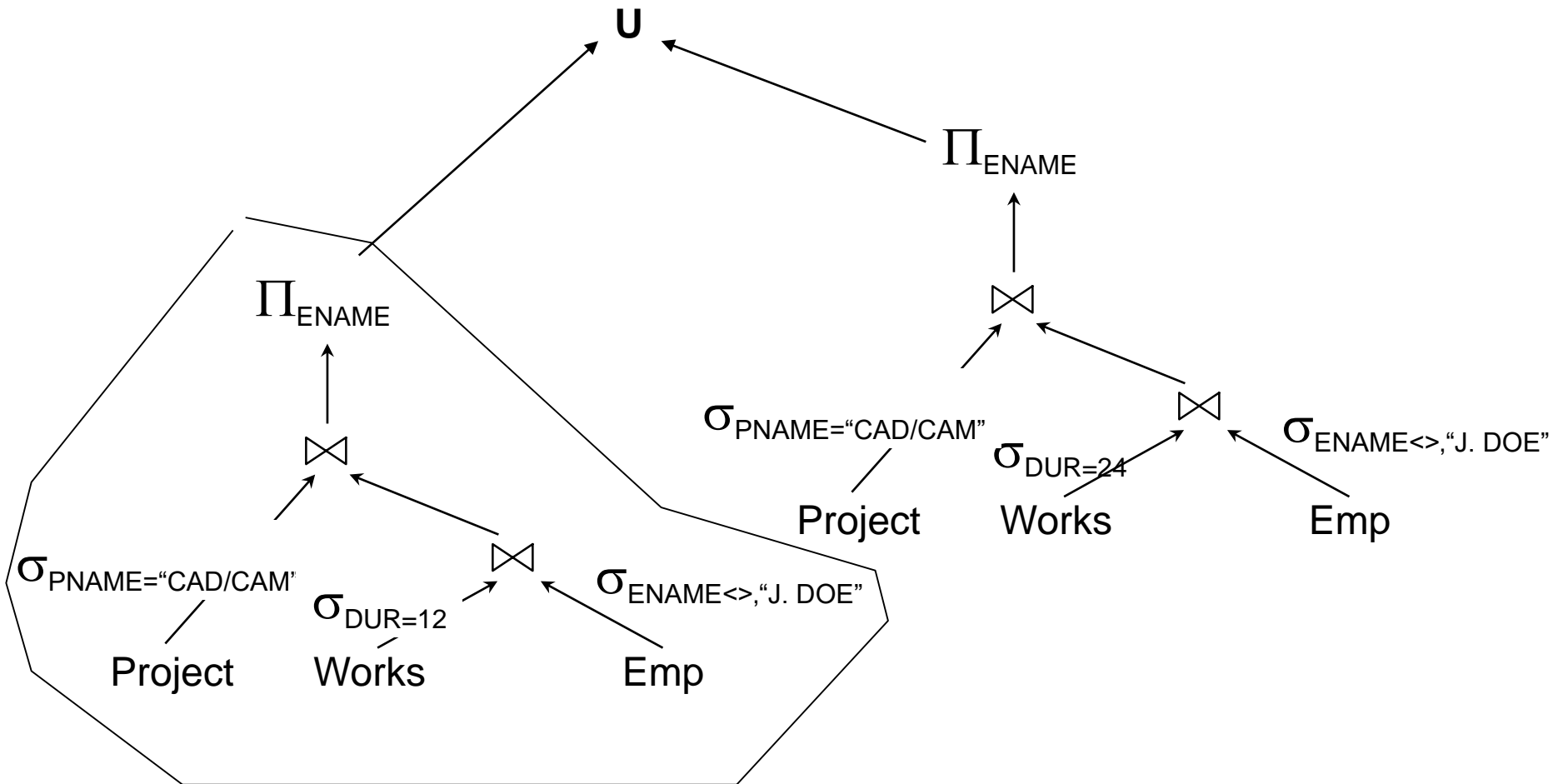


Autre plan équivalent

Plus efficace que les plans précédents ?



Encore un autre plan équivalent

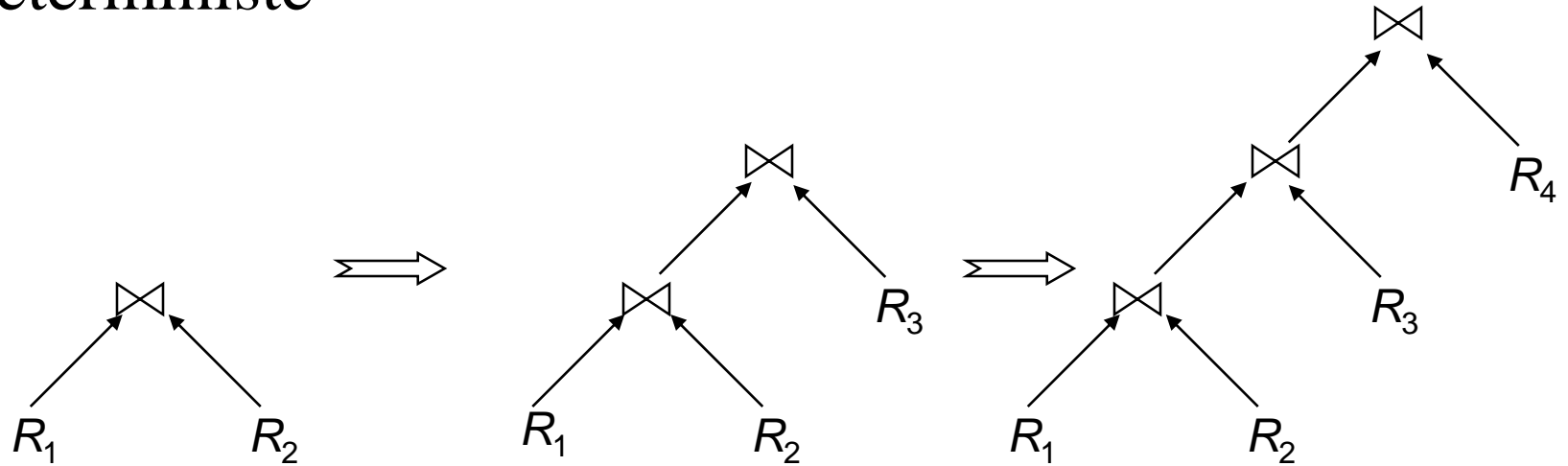


Stratégie de recherche

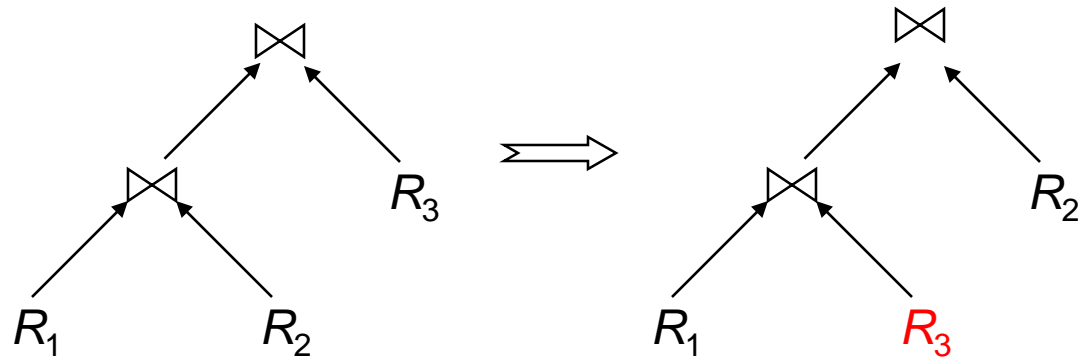
- Il est en général trop coûteux de faire une recherche exhaustive
- Déterministe
 - part des relations de base et construit les plans en ajoutant une relation à chaque étape
 - programmation dynamique: largeur-d'abord
 - excellent jusqu'à 5-6 relations
- Aléatoire
 - recherche l'optimalité autour d'un point de départ particulier
 - réduit le temps d'optimisation (au profit du temps d'exécution)
 - meilleur avec $> 5-6$ relations
 - recuit simulé (simulated annealing)
 - amélioration itérative (iterative improvement)

Stratégies de recherche

- Déterministe



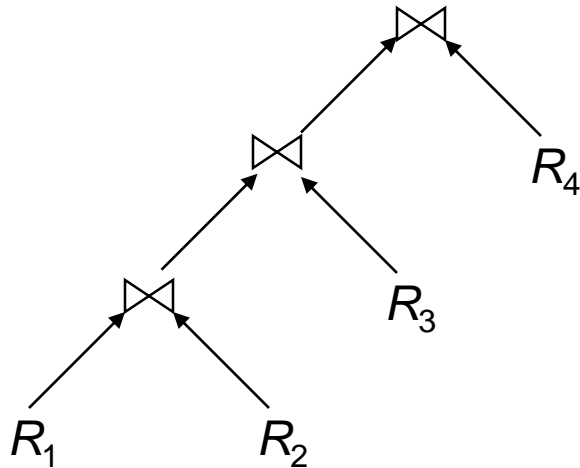
- Aléatoire



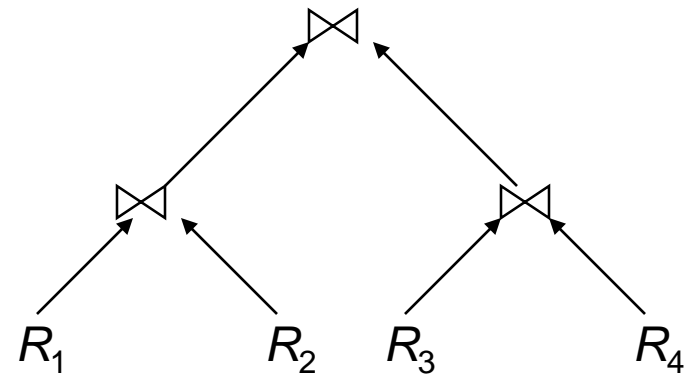
Algorithmes de recherche

- Limiter l'espace de recherche
 - heuristiques
 - par ex. appliquer les opérations unaires avant les autres
 - Ne marche pas toujours (perte d'index, d'ordre)
 - limiter la forme des arbres

Arbre linéaire



Arbre touffu



Génération de plan physique

- Sélection :
 - Commencer par les conditions d'égalité avec un index sur l'attribut
 - Filtrer sur cet ensemble de n-uplets ceux qui correspondent aux autres conditions
- Jointure
 - Utilisation des index, des relations déjà triées sur l'attribut de jointure, présence de plusieurs jointures sur le même attribut
- Pipelines ou matérialisation

Conclusion

- Point fondamental dans les SGBD
- Importance des métadonnées, des statistiques sur les relations et les index, du choix des structures d'accès.
- L'administrateur de bases de données peut améliorer les performances en créant de nouveaux index, en réglant certains paramètres de l'optimiseur de requêtes (voir TME)

Conclusion

- Un SGBD doit transformer une requête déclarative en un programme impératif :
 - Plan d'exécution
 - Algèbre
- Calculer les tailles des résultats intermédiaire donne une idée du coût d'un plan mais..
 - Comment mettre en œuvre les opérateurs ?
 - Comment enchaîner les opérateurs ?
 - Comment trouver le meilleur plan en fonction de ce qui précède

Réponses dans les
prochains cours