Master d'Informatique
spécialité DAC
BDLE (Bases de Données Large Echelle)
-Seconde Partie-

Cours 3 : Requêtes relationnelles en M/R (2/2)

Mohamed-Amine Baazizi – email: prénom.nom@lip6.fr
http://dac.lip6.fr/master/ues-2014-2015/bdle-2014-2015/

Objectifs

Cours

Traduction des requêtes relationnelles SQL

TME

1. Jointures parallèles sur le cluster
2. Traduction des requêtes SQL

2

Bilan cours précédent

Traduction d'opérateurs algébriques

- Algorithmes facilement généralisables
- Jointures n-aires réalisables en une seule passe, coût polynomial.

Exemple :

$Q = R(A,B) \bowtie S(B,C) \bowtie T(C,D)$

coût(Q) $= |S| + |D_C| \times |R| + |D_B| \times |T|$

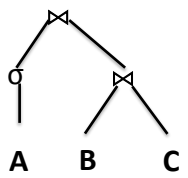
où D_B et D_C domaines fonctions hachage sur B et C

3

SQL (version simplifiée)

Opérateurs algébriques + fonctions d'agrégats

- Sélection, projection, jointure
- COUNT, SUM, AVG, MIN, MAX



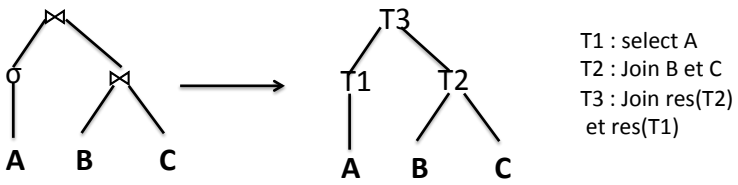
Exemple d'un arbre algébrique

4

Traduction des requêtes SQL

Naïve (one-to-one)

- Chaque nœud = une tâche MR
- Une requête = enchaînement de tâches MR



T1 : select A
T2 : Join B et C
T3 : Join res(T2)
et res(T1)

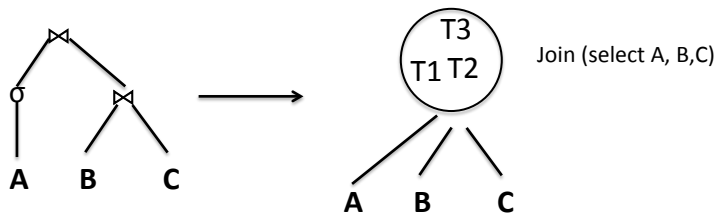
matérialisation + chargement résultats intermédiaires
➔ Coût élevé

5

Traduction des requêtes SQL

Optimisée

- Chaque groupe de nœuds = une tâche MR
- Une requête = enchaînement de tâches MR



Join (select A, B,C)

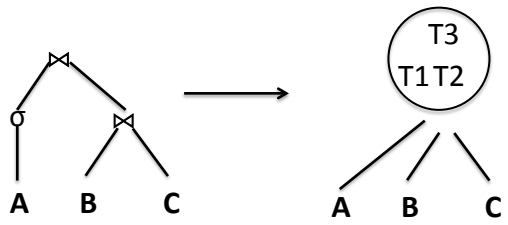
Pas de matérialisation/chargement inutile
➔ Coût réduit

6

Traduction optimisée

Principe : combiner les opérateurs bottom-up

- Jointures seules : cours précédent
- Jointures avec opérateurs : en général toujours possible excepté agrégation ou tri



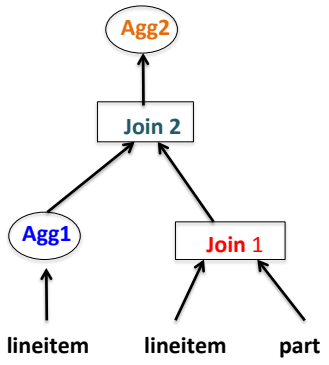
Join (select A, B,C)

7

Traduction optimisée : illustration

```
SELECT sum(l_extendedprice) / 7.0 AS
avg_yearly
FROM (SELECT l_partkey, 0.2*
avg(l_quantity) AS t1
      FROM lineitem GROUP BY
            l_partkey) AS inner,
     (SELECT
     l_partkey,l_quantity,l_extendedprice
     FROM lineitem, part
     WHERE p_partkey = l_partkey) AS
                outer
WHERE outer.l_partkey =
inner.l_partkey; AND outer.l_quantity <
inner.t1;
```

Q17 de TPC-H



Workflow MR

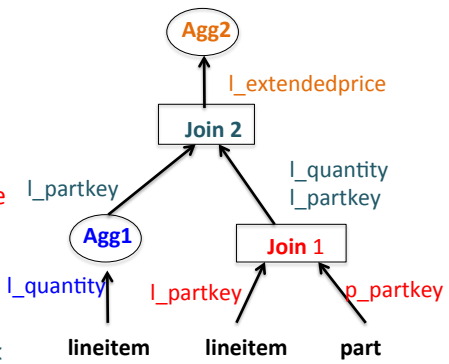
Adapté de [Lee]

8

Traduction optimisée : illustration

```
SELECT sum(l_extendedprice) / 7.0 AS
avg_yearly
FROM (SELECT l_partkey, 0.2*
avg(l_quantity) AS t1
      FROM lineitem GROUP BY
            l_partkey) AS inner,
     (SELECT
     l_partkey,l_quantity,l_extendedprice
     FROM lineitem, part
     WHERE p_partkey = l_partkey) AS
                outer
WHERE outer.l_partkey =
inner.l_partkey; AND outer.l_quantity <
inner.t1;
```

Q17 de TPC-H



Workflow MR

Adapté de [Lee]

9

Traduction optimisée : illustration

Agg1

Map : lineitem→(l_partkey,l_quantity)

Reduce : 0.2* **avg =v**

Join1

Map : lineitem→(l_partkey, (l_quantity, l_extendedprice))

part→(l_partkey, null)

Reduce : jointure

Join2

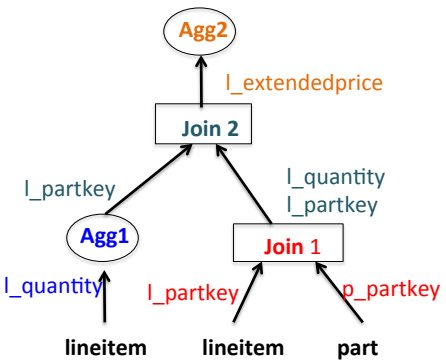
Map : inner→(l_partkey,**v**)

outer →(l_partkey,(l_quantity , l_extendedprice))

Reduce : jointure

Agg2 …

Programmes MR



Workflow MR

Adapté de [Lee]

10

Traduction optimisée : illustration

Join

Map :

lineitem→(p_partkey , (l_quantity l_extendedprice))

part→(l_partkey, null)

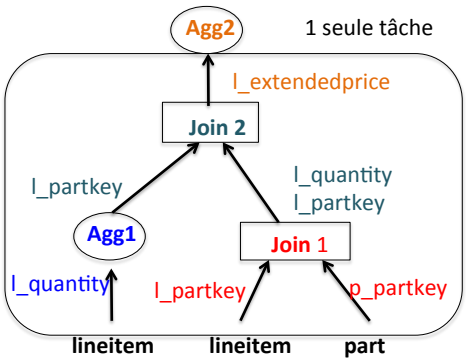
Reduce :

agg1 sur l_quantity

join1 sur l_partkey = p_partkey

join2 sur l_partkey = l_partkey

Programme MR



Coût communication --

Workflow MR

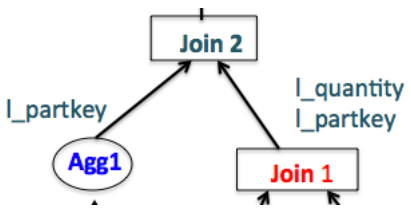
Adapté de [Lee]

11

Traduction optimisée : intuition

Résultat d'une tâche = entrée de la suivante

Même(s) attribut(s) de partitionnement → *transit correlation (TC)*

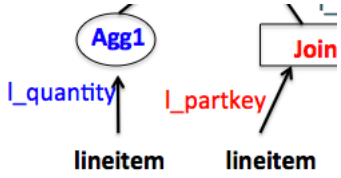


12

Traduction optimisée : intuition

Auto-jointures

Mêmes données initiales → *Input correlation (IC)*

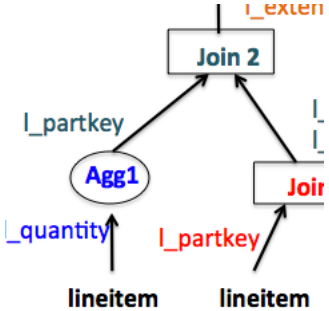


13

Traduction optimisée : intuition

Résultat d'une tâche = entrée de la suivante

Mêmes données initiales + même(s) attribut(s) de partitionnement →*Jobflow correlation (JFC)*



14

Traduction optimisée : gain

Input correlation	Partager les Map	• Gain local (accès disque) • communication si map distant
Transit correlation	Partager Map, mutualiser reduces	• Gain local (accès disque) • communication
Jobflow correlation	Mutualiser reduce	• Gain local (accès disque) • communication

15

Traduction optimisée : gain

Join

Map :

lineitem→(p_partkey , (l_quantity l_extendedprice))

part→(l_partkey, null)

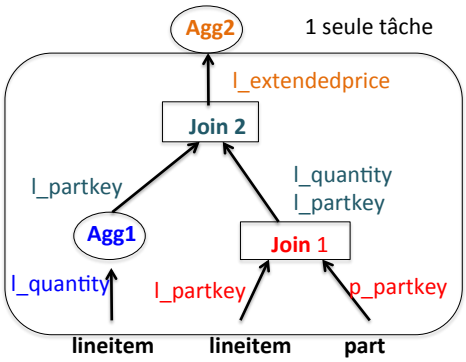
Reduce :

agg1 sur l_quantity

join1 sur l_partkey = p_partkey

join2 sur l_partkey = l_partkey

Programme MR



Workflow MR

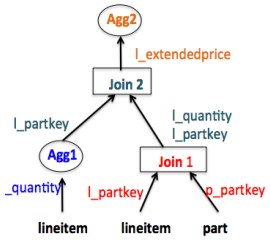
Adapté de [Lee]

Comparer coût communication des approches naive et optim

16

Traduction optimisée : aperçu

SQL → **MR^{std}** → **MR^{optim}**



Workflow MR

Agg1
Map : lineitem→(l_partkey, l_quantity)
Reduce : 0.2* **avg =v**

Join1
Map : lineitem→(l_partkey, (l_quantity, l_extendedprice))
part→(l_partkey, null)
Reduce : jointure

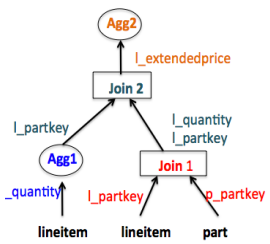
Join2
Map : inner→(l_partkey,**v**)
outer →(l_partkey,(l_quantity , l_extendedprice))
Reduce : jointure

Agg2 …

Map :
lineitem→(p_partkey , (l_quantity l_extendedprice))
part→(l_partkey, null)
Reduce :
agg1 sur l_quantity
join1 sur l_partkey = p_partkey
join2 sur l_partkey = l_partkey

Traduction optimisée : aperçu

SQL → **MR^{std}** → **MR^{optim}**



Workflow MR

Proj-sel-Job *
Agg-Job
Join-job**
Sort-job

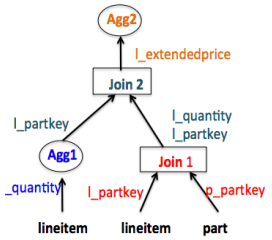
4 types de tâches Pré-optimisées

Map :
lineitem→(p_partkey , (l_quantity l_extendedprice))
part→(l_partkey, null)
Reduce :
agg1 sur l_quantity
join1 sur l_partkey = p_partkey
join2 sur l_partkey = l_partkey

* Seulement pour des requêtes sans join/agg (inutiles en pratique)
** Map prenant en charge proj, sel et agg

Traduction optimisée : aperçu

SQL → MR^std → MR^optim



Workflow MR

Proj-sel-Job *
Agg-Job
Join-job**
Sort-job

4 types de tâches Pré-optimisées

Map :
lineitem→(p_partkey , (l_quantity l_extendedprice))
part→(l_partkey, null)
Reduce :
agg1 sur l_quantity
join1 sur l_partkey = p_partkey
join2 sur l_partkey = l_partkey

* Seulement pour des requêtes sans join/agg (inutiles en pratique)
** Map prenant en charge proj, sel et agg

19

Traduction optimisée : étapes

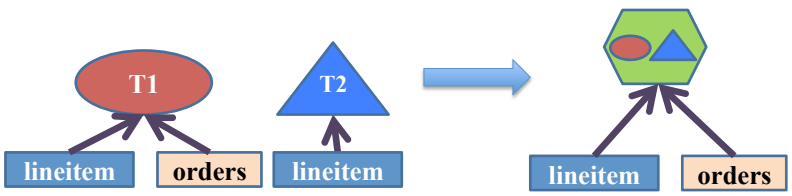
A. Aplatir l'arbre MR^std : parcours post-order
   → séquence de tâches (PS-A-J-S)

B. Réduire la séquence de tâches
   1. Fusionner des opérateurs
      a. (SP, Agg, Sort, Join) x (SP, Agg, Sort)
      b. Jointures successives
   2. Agrégation au vol

20

Règles de transformation

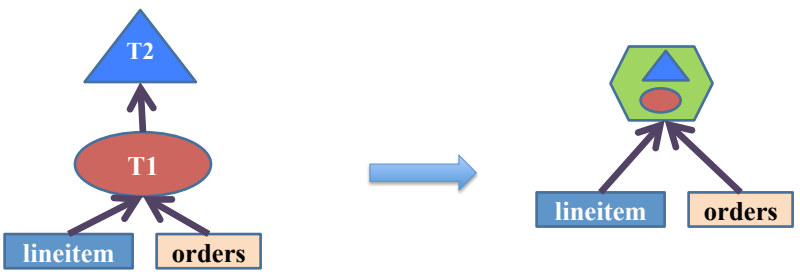
Règle 1 $\dfrac{\text{IC}(T1,T2) \quad \text{TC}(T1,T2) \quad T12(s)=T1(s)\text{o}T2(s)\ s \text{ in } \{\text{Map, Reduce}\}}{T12}$



21

Règles de transformation

Règle 2 $\dfrac{\text{JFC}(T1,T2) \ \ \text{parent}(T2,T1) \quad T1 : \text{agg} \quad T12(\text{Reduce})=\text{agg}}{T12}$



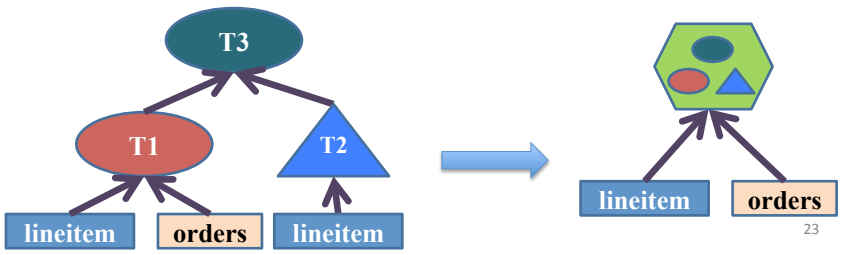
22

Règles de transformation

JFC(T3,T1) JFC(T3,T2) TC(T1,T2) T1<T2<T3
T123(s)=T1(s)oT2(s)oT3(s) s in {Map, Reduce}

Règle 3 ——————————

T123



23

23

Déclenchement des règles

Phase 1 : appliquer règle 1 (fusion) jusqu'à plus de tâches IC et TC

Phase 2 : appliquer les règles 2 à 3 (pour JFC)

- Règle 2 : Agrégation
- Règle 3 : JFC avec les deux précédentes tâches

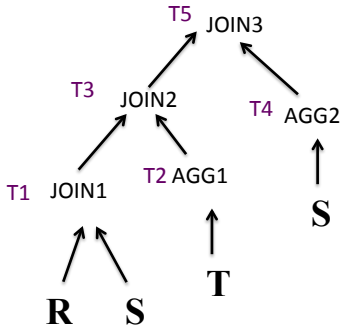
24

Exemples

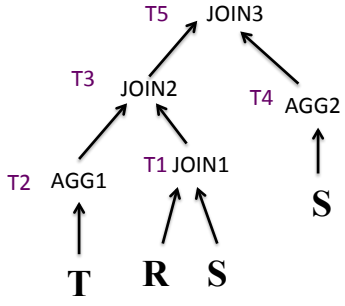
R(A,B) S(B,C) et T(C,D) trois schémas de relations

Q = R ⋈ S ⋈ agg(T) ⋈ agg(S)

Les corrélations : TC(T1,T4), JFC(T3,T1), JFC(T5,T3), JFC(T5,T4)



Résultat : T14, T2, T35
(3 tâches)



Résultat : T2, T1435
(2 tâches)

25

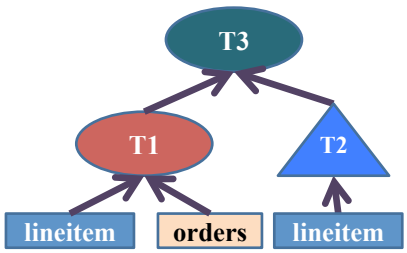
Règles de transformation

JFC(T3,T1) T1<T2<T3

T13(Reduce)=T1(reduce)oT3(reduce)

Règle 4 ——————————————

T2 , T13



Attention : aucune corrélation entre T3 et T2!

1. **T2 s'éxecute**
2. **T3 exécutée dans le reduce de T1**

26

Déclenchement des règles

Phase 1 : appliquer règle 1 (fusion) jusqu'à plus de tâches IC et TC

Phase 2 : appliquer les règles 2 à 3 (pour JFC)

- Règle 2 : Agrégation
- Règle 3 : JFC avec les deux précédentes tâches
- Règle 4 : JFC avec une seule tâche précédente

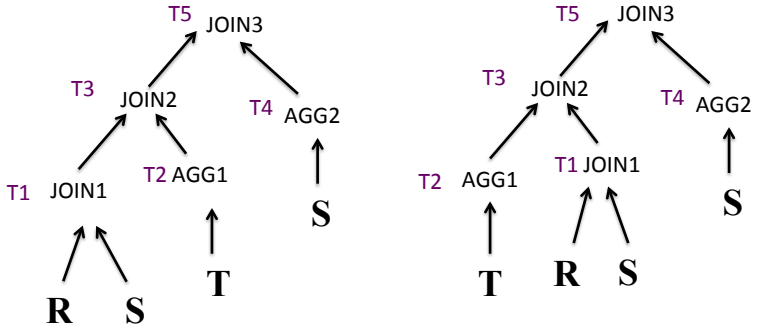
27

Exemples

R(A,B) S(B,C) et T(C,D) trois schémas de relations

Q = R ⋈ S ⋈ agg(T) ⋈ agg(S)

Les corrélations : TC(T1,T4), JFC(T3,T1), JFC(T5,T3), JFC(T5,T4)



Même résultat : T2, T1435
(2 tâches)

28

Références

[Lee et al.] *YSmart: Yet Another SQL-to-MapReduce Translator*, in ICDCS'2011

[YSmart] https://github.com/YSmart/YSmart

29