Rapport du TP2 : Performances et Tuning pour le traitement de Masses de Données

Gestion de Grandes Masses de Données (GGMD)

par NGUYEN Cécilia et TANG Kévin

fait le 8 Octobre 2023

Université Claude Bernard Lyon 1

Table des matières

1. Les requêtes

Requête Q1

```
SELECT CASE WHEN lieudeces LIKE '97%' THEN SUBSTRING(lieudeces, 0, 4) ELSE SUBSTRING(lieudeces, 0, 3) END as numdep, COUNT(*) AS nbD FROM
   (
       SELECT nomprenom, datenaiss, lieunaiss, datedeces, lieudeces, COUNT(*) as nbD
       FROM personne_insee_medium
       GROUP BY nomprenom, datenaiss, lieunaiss, datedeces, lieudeces
       HAVING COUNT(*) > 1
   )
GROUP BY numdep
ORDER BY COUNT(*) DESC;
```

Requête Q2

```
 WITH SS1 AS (
  SELECT
    generate_series('1970-01-01'::date, CURRENT_DATE, '1 decade'::interval) AS decades_series
),
SS2 AS (
  SELECT
    split_part(nomprenom, '*', 1) AS nom,
    EXTRACT(YEAR FROM TO_DATE(datenaiss, 'YYYYMMDD'))::integer AS annee_naissance,
    EXTRACT(YEAR FROM TO_DATE(datedeces, 'YYYYMMDD'))::integer AS annee_deces
  FROM
    personne_insee_medium
  WHERE
    is_date(datenaiss) AND is_date(datedeces)
),
RankedNames AS (
  SELECT
    EXTRACT(YEAR FROM SS1.decades_series) AS decades,
    SS2.nom,
    COUNT(*) AS occurrences,
    ROW_NUMBER() OVER (PARTITION BY EXTRACT(YEAR FROM SS1.decades_series) ORDER BY COUNT(*) DESC) AS rank
  FROM SS1
  JOIN SS2 ON
    EXTRACT(YEAR FROM SS1.decades_series) BETWEEN SS2.annee_naissance AND SS2.annee_deces
  GROUP BY SS1.decades_series, SS2.nom
)
SELECT
  decades,
  STRING_AGG(rank || '. ' || nom || ' (' || occurrences || ')', ' ; ') AS classement
FROM RankedNames
WHERE rank <= 10
GROUP BY decades
ORDER BY decades;
```

Requête Q3

```
 WITH Personnes AS (
    SELECT
        nomprenom,
        CASE
            WHEN SUBSTRING(lieunaiss FROM 1 FOR 2) = '97' THEN SUBSTRING(lieunaiss FROM 1 FOR 3)
            ELSE SUBSTRING(lieunaiss FROM 1 FOR 2)
        END AS dep_naissance,
            CASE
            WHEN SUBSTRING(lieudeces FROM 1 FOR 2) = '97' THEN SUBSTRING(lieudeces FROM 1 FOR 3)
            ELSE SUBSTRING(lieudeces FROM 1 FOR 2)
        END AS dep_deces,
        (TO_DATE(datedeces, 'YYYYMMDD') - TO_DATE(datenaiss, 'YYYYMMDD')) AS duree_vie
    FROM
        personne_insee_medium
    WHERE
            is_date(datenaiss) AND is_date(datedeces)
)
SELECT
    d.dep,
    d.nom,
    COUNT(*) AS nombre_personnes_deces,
    justify_interval( (AVG(duree_vie) || ' days')::interval ) AS esperance_vie_moyenne
FROM Personnes p
JOIN departement d ON p.dep_naissance = d.dep
GROUP BY d.dep, d.nom
ORDER BY esperance_vie_moyenne DESC, d.dep;
```

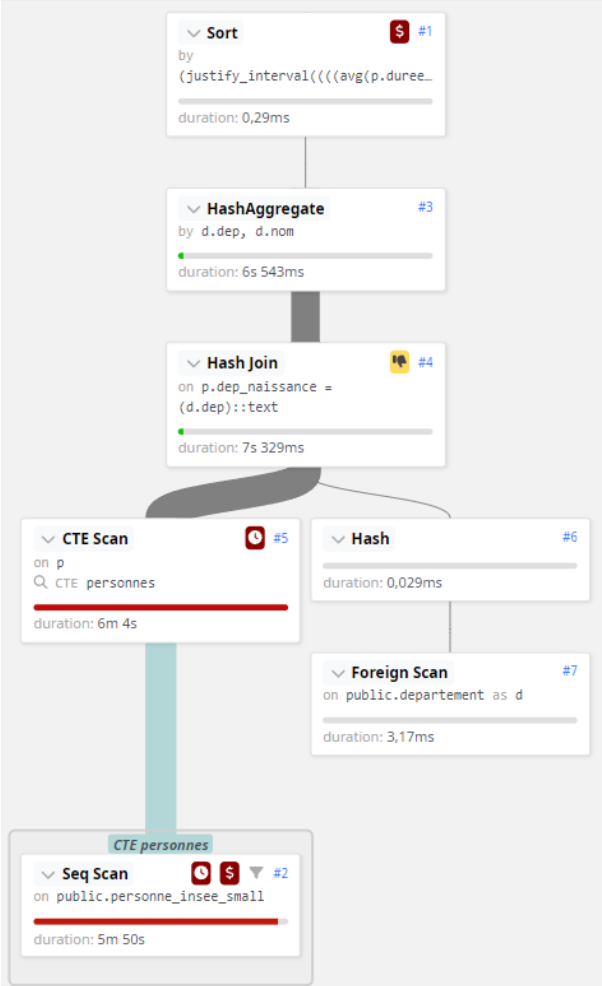
2. Plans d’exécution

a. Traitement des requêtes sur GGMD1

Q1, Q2 / GGMD1 / Aucun traitement

Les plans d’exécution des requêtes Q1 et Q2 n’ont pas pu aboutir car la VM GGMD1 dispose de trop peu de mémoire.
Une idée serait de créer des index qui faciliteront l’exécution des requêtes.

Q3 / GGMD1 / Aucun traitement - [Lien Dalibo]



Per table stats

Table	Count	Time	
public.personne_insee_small	1	5m 50s	92%
#2 Seq Scan		5m 50s	92%
public.departement	1	3,17ms	0%
#7 Foreign Scan		3,17ms	0%

Per node type stats

Node Type	Count	Time	
CTE Scan	1	6m 4s	96%
#5 CTE Scan		6m 4s	96%
Seq Scan	1	5m 50s	92%
#2 Seq Scan		5m 50s	92%
Hash Join	1	7s 329ms	2%
#4 Hash Join		7s 329ms	2%
HashAggregate	1	6s 543ms	2%
#3 HashAggregate		6s 543ms	2%
Foreign Scan	1	3,17ms	0%
#7 Foreign Scan		3,17ms	0%
Sort	1	0,29ms	0%
#1 Sort		0,29ms	0%
Hash	1	0,029ms	0%
#6 Hash		0,029ms	0%

Per index stats

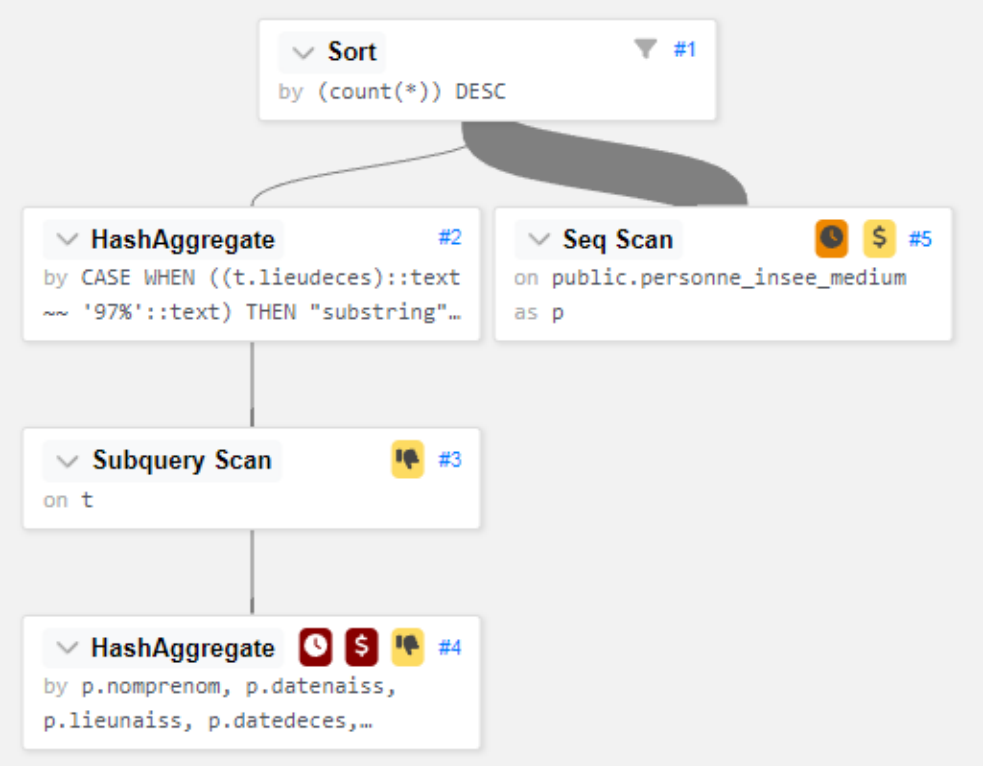
Index	Count	Time
No index used		

Q1, Q2, Q3 / GGMD1 / Index

Aucun index n’a pu être créé sur la VM GGMD1 par manque de mémoire. De ce fait, aucun plan d’exécution avec des index n’a pu être récupéré sur GGMD1.

b. Traitement des requêtes sur GGMD2

Q1 / GGMD2 / Aucun traitement - [Lien Dalibo]



Per table stats

Table	Count	Time	
public.personne_insee_medium	1	3m 14s	56%

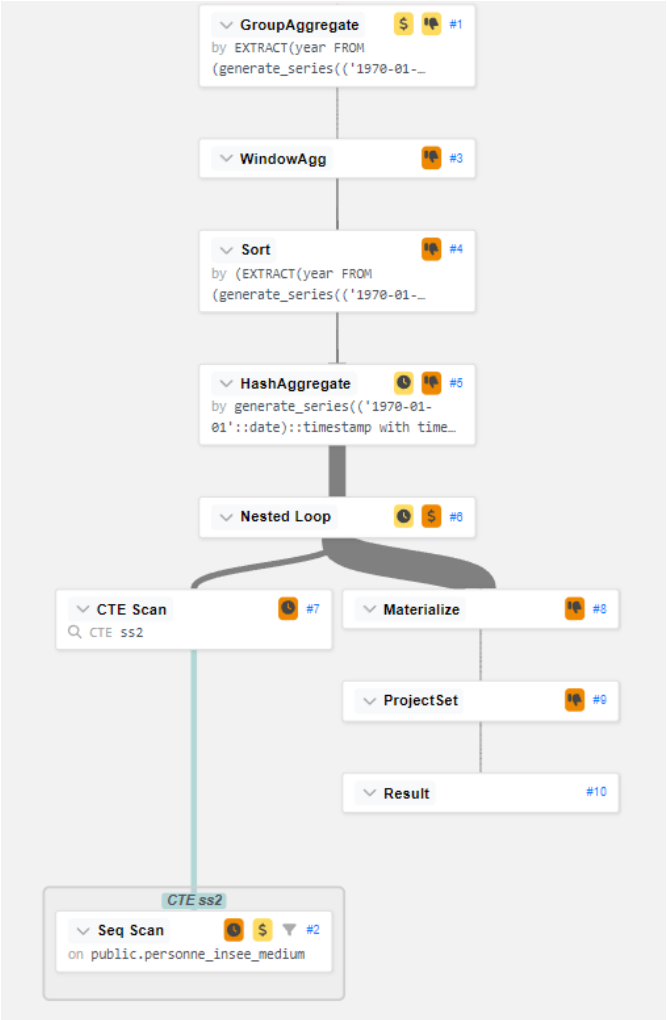
Per node type stats

Node Type	Count	Time	
HashAggregate	2	5m 44s	100%
Seq Scan	1	3m 14s	56%
Subquery Scan	1	41,9ms	0%
Sort	1	0ms	0%

Per index stats

Index	Count	Time
No index used		

Q2 / GGMD2 / Aucun traitement - [Lien Dalibo]



Per table stats

Table	Count	Time	
public.personne_insee_medium	1	7m 34s	58%

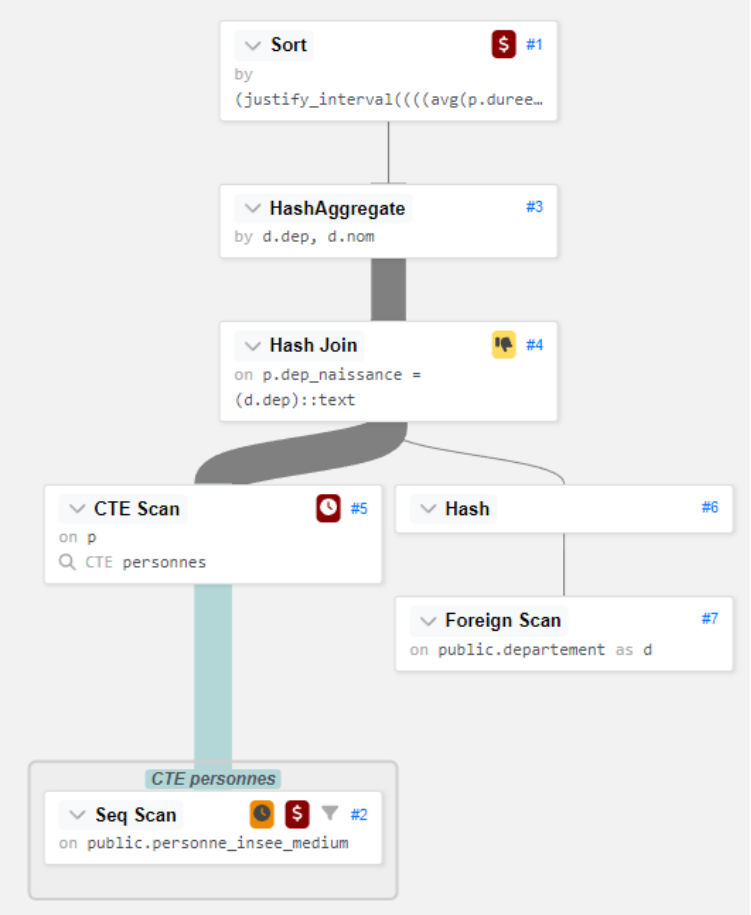
Per node type stats

Node Type	Count	Time	
CTE Scan	1	7m 46s	60%
Seq Scan	1	7m 34s	58%
HashAggregate	1	3m 36s	28%
Nested Loop	1	1m 29s	11%
Sort	1	5s 24,1ms	1%
WindowAgg	1	816ms	0%
GroupAggregate	1	0,282ms	0%
ProjectSet	1	0,098ms	0%
Result	1	0,017ms	0%
Materialize	1	0ms	0%

Per index stats

Index	Count	Time
No index used		

Q3 / GGMD2 / Aucun traitement - Lien Dalibo



Per table stats

Table	Count	Time	
public.personne_insee_medium	1	3m 9s	88%
public.departement	1	2,51ms	0%

Per node type stats

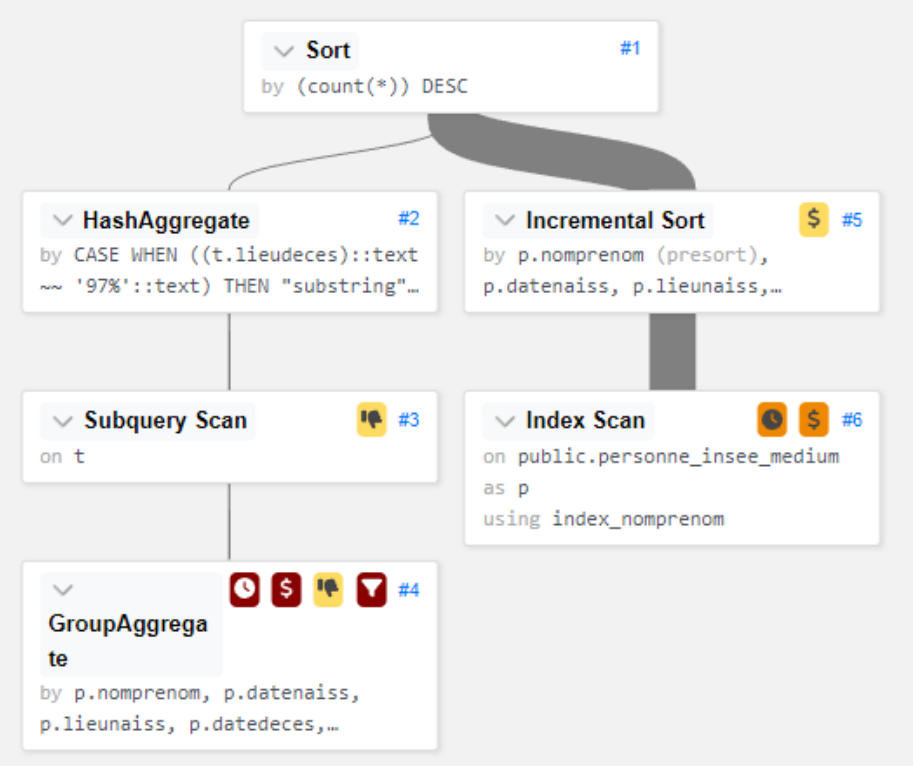
Node Type	Count	Time	
CTE Scan	1	3m 21s	94%
Seq Scan	1	3m 9s	88%
Hash Join	1	6s 487ms	3%
HashAggregate	1	6s 108ms	3%
Foreign Scan	1	2,51ms	0%
Sort	1	0,104ms	0%
Hash	1	0,062ms	0%

Per index stats

Index	Count	Time
No index used		

Q1 / GGMD2 / Index - Lien Dalibo

Index utilisé : *nomprenom* de personne_insee.



Per table stats

Table	Count	Time	
public.personne_insee_medium	1	2m 9s	87%

Per node type stats

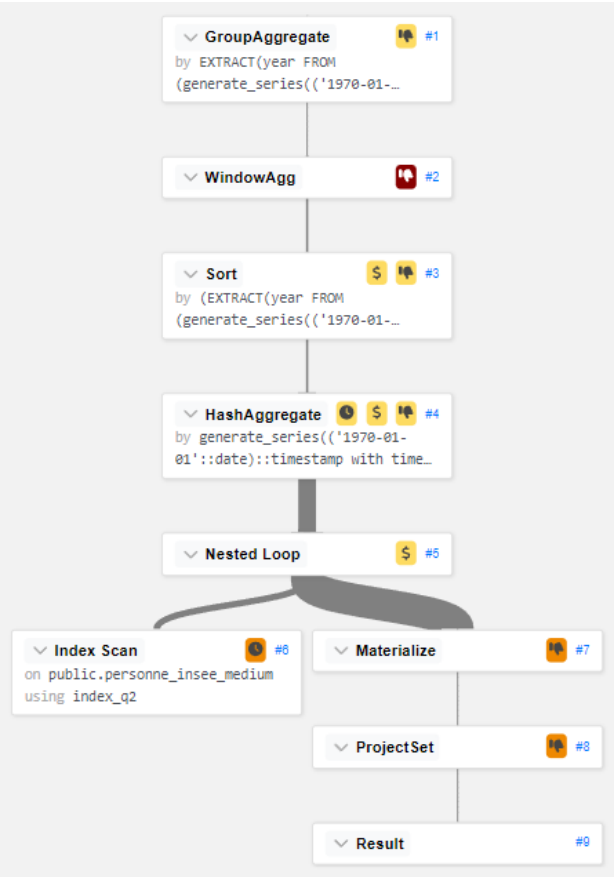
Node Type	Count	Time	
GroupAggregate	1	2m 28s	100%
Index Scan	1	2m 9s	87%
Incremental Sort	1	10s 23,6ms	7%
HashAggregate	1	141ms	0%
Subquery Scan	1	111ms	0%
Sort	1	0ms	0%

Per index stats

Index	Count	Time	
index_nomprenom	1	2m 9s	87%

Q2 / GGMD2 / Index - Lien Dalibo

Index utilisé : datenaiss et datedeces de personne_insee.



Per table stats

Table	Count	Time	
public.personne_insee_medium	1	1h 11m	82%

Per node type stats

Node Type	Count	Time	
Index Scan	1	1h 11m	82%
HashAggregate	1	10m 40s	12%
Nested Loop	1	4m 46s	5%
Sort	1	15s 241ms	0%
WindowAgg	1	494ms	0%
Result	1	316ms	0%
GroupAggregate	1	0,132ms	0%
ProjectSet	1	0,054ms	0%
Materialize	1	0ms	0%

Per index stats

Index	Count	Time	
index_q2	1	1h 11m	82%

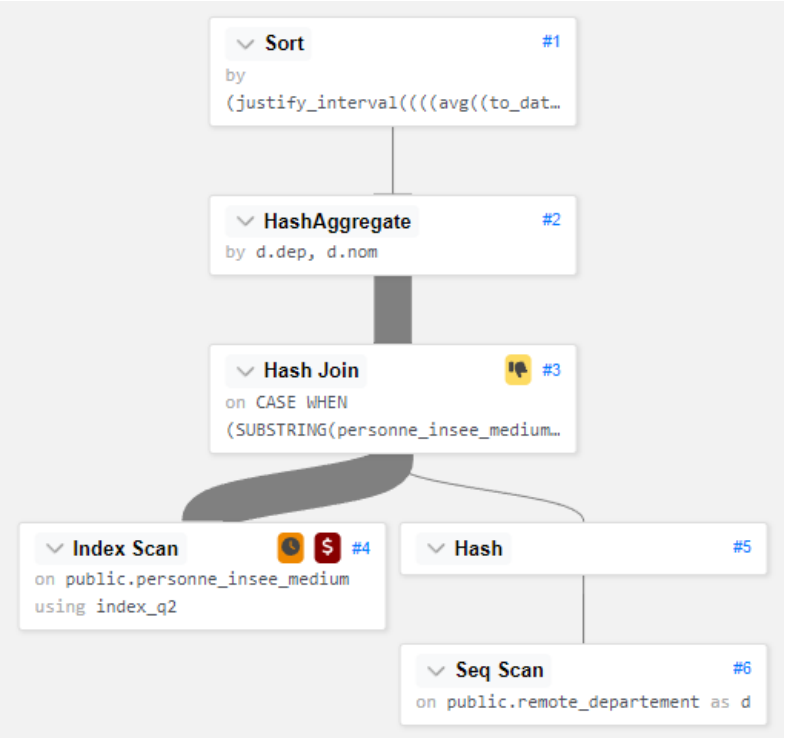
Test réalisé le 08/10, la VM était particulièrement lente.
Les jours précédents, les requêtes effectuées prenaient environ 10 minutes à être traitées. Celle-ci ainsi que toutes les requêtes qui ont suivi ont toutes pris une ou plusieurs heures à s'exécuter. Ce qui est problématique car cela faussera les résultats des graphiques de synthèse.
Concernant le coût de la requête, il reste sensiblement le même sur les différentes VM.

Nous n'avons donc pas ajouté les temps dans le graphique de synthèse, voici les résultats obtenus :

- Temps de planification : 688.429 ms
- Temps d'exécution : 5247941.534 ms

Q3 / GGMD2 / Index - Lien Dalibo

Index utilisé : datenaiss et datedeces de personne_insee.



Per table stats

Table	Count	Time	
public.personne_insee_medium	1	2m 56s	85%
public.remote_departement	1	280ms	0%

Per node type stats

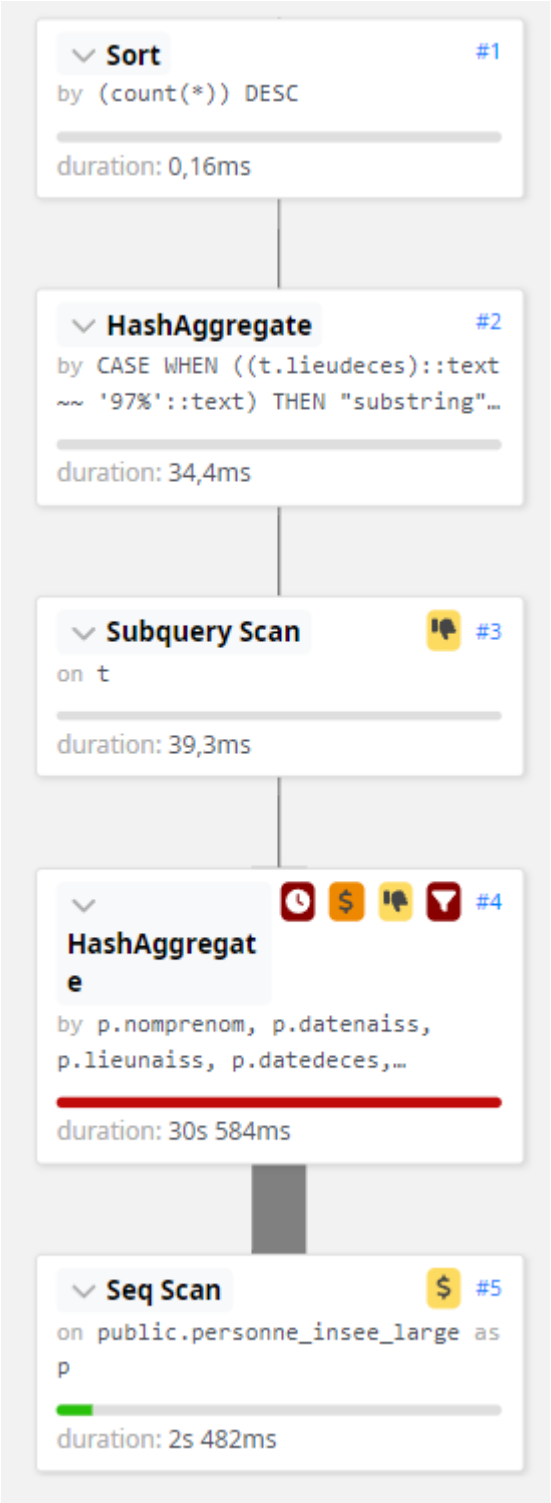
Node Type	Count	Time	
Index Scan	1	2m 56s	85%
HashAggregate	1	17s 962ms	9%
Hash Join	1	12s 854ms	6%
Seq Scan	1	280ms	0%
Sort	1	0,197ms	0%
Hash	1	0,06ms	0%

Per index stats

Index	Count	Time	
index_q2	1	2m 56s	85%

c. Traitement des requêtes sur GGMD3

Q1 / GGMD3 / Aucun traitement - [Lien Dalibo]



Per table stats

Table	Count	Time	
public.personne_insee_large	1	2s 482ms	7%
#5 Seq Scan		2s 482ms	7%

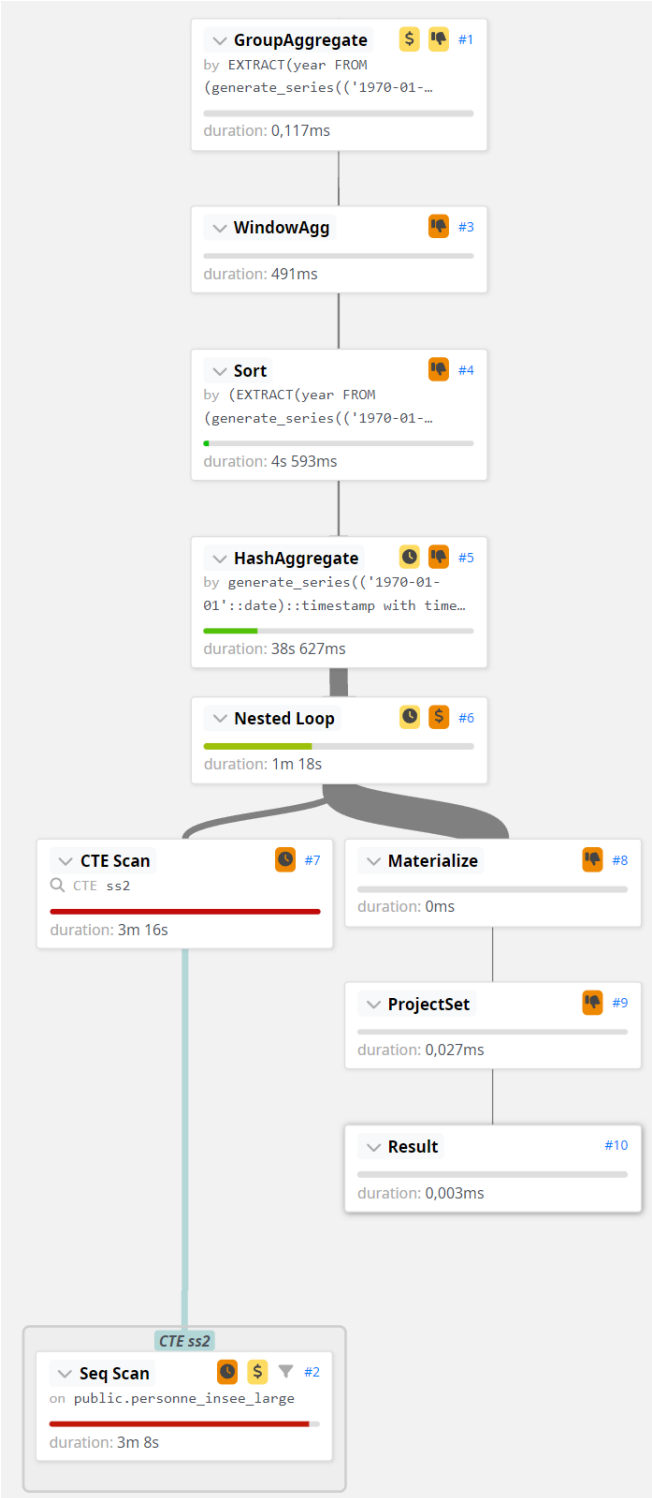
Per node type stats

Node Type	Count	Time	
HashAggregate	2	30s 618ms	92%
#4 HashAggregate		30s 584ms	92%
#2 HashAggregate		34,4ms	0%
Seq Scan	1	2s 482ms	7%
#5 Seq Scan		2s 482ms	7%
Subquery Scan	1	39,3ms	0%
#3 Subquery Scan		39,3ms	0%
Sort	1	0,16ms	0%
#1 Sort		0,16ms	0%

Per index stats

Index	Count	Time
No index used		

Q2 / GGMD3 / Aucun traitement - [Lien Dalibo]



Per table stats

Table	Count	Time	
public.personne_insee_large	1	3m 8s	59%

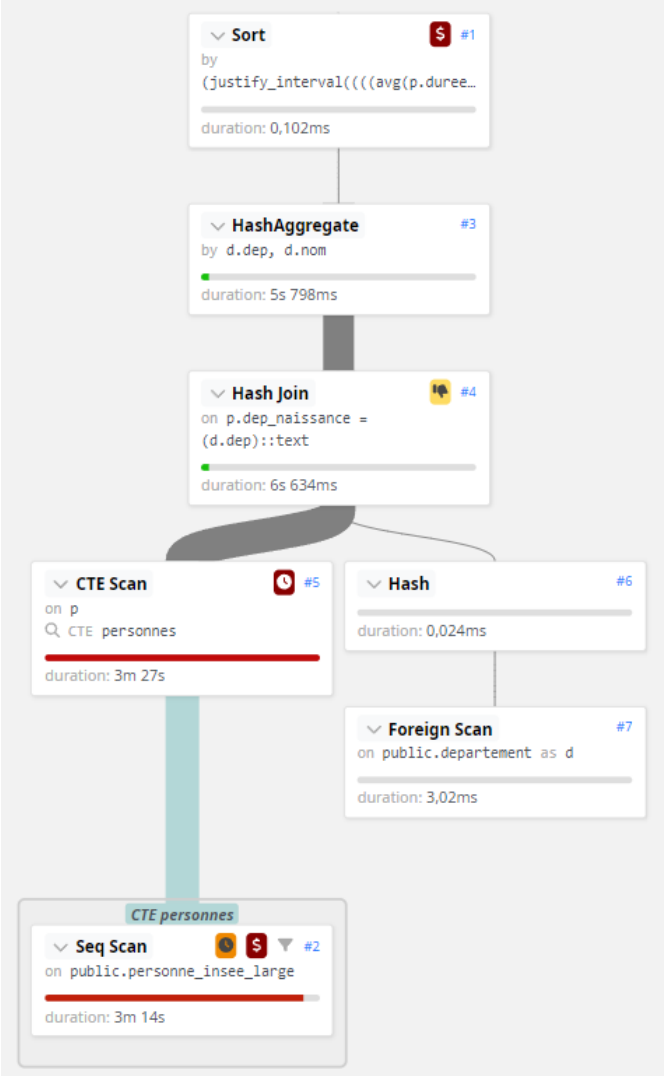
Per node type stats

Node Type	Count	Time	
CTE Scan	1	3m 16s	62%
#7 CTE Scan		3m 16s	62%
Seq Scan	1	3m 8s	59%
#2 Seq Scan		3m 8s	59%
Nested Loop	1	1m 18s	25%
#6 Nested Loop		1m 18s	25%
HashAggregate	1	38s 627ms	12%
#5 HashAggregate		38s 627ms	12%
Sort	1	4s 593ms	1%
#4 Sort		4s 593ms	1%
WindowAgg	1	491ms	0%
#3 WindowAgg		491ms	0%
GroupAggregate	1	0,117ms	0%
#1 GroupAggregate		0,117ms	0%
ProjectSet	1	0,027ms	0%
#9 ProjectSet		0,027ms	0%
Result	1	0,003ms	0%
#10 Result		0,003ms	0%
Materialize	1	0ms	0%
#8 Materialize		0ms	0%

Per index stats

Index	Count	Time
No index used		

Q3 / GGMD3 / Aucun traitement - [Lien Dalibo]



Per table stats

Table	Count	Time	
public.personne_insee_large	1	3m 14s	88%
#2 Seq Scan		3m 14s	88%
public.departement	1	3,02ms	0%
#7 Foreign Scan		3,02ms	0%

Per node type stats

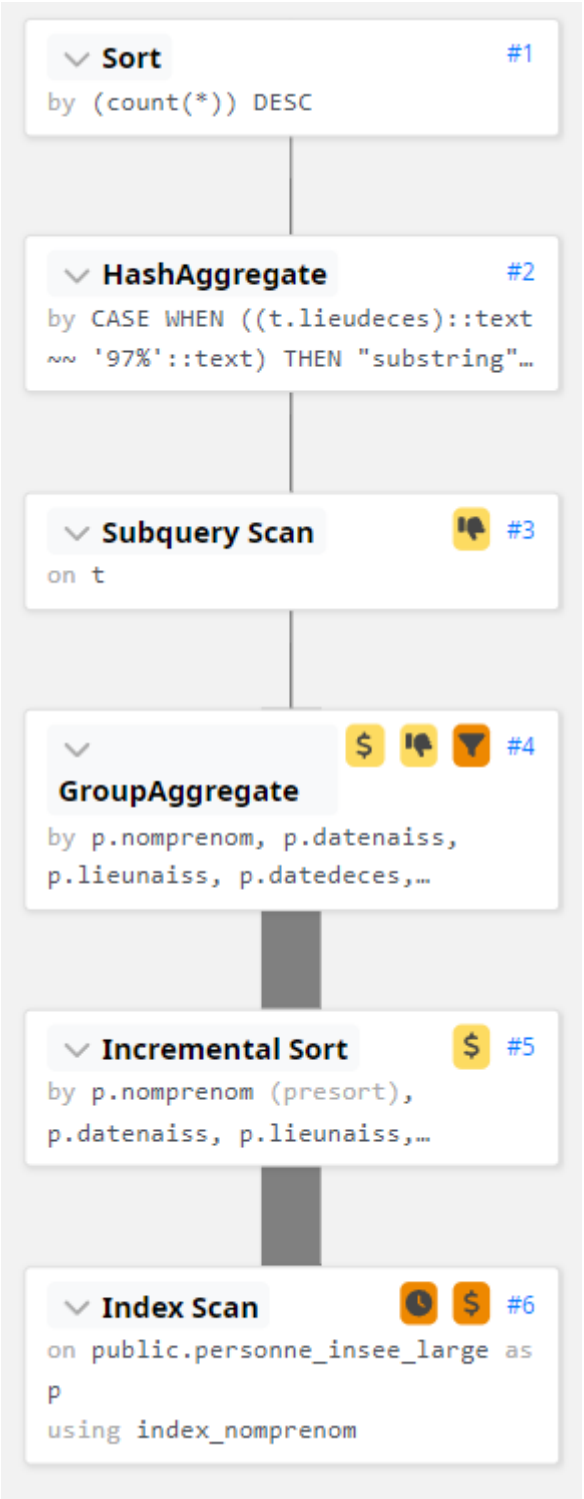
Node Type	Count	Time	
CTE Scan	1	3m 27s	94%
#5 CTE Scan		3m 27s	94%
Seq Scan	1	3m 14s	88%
#2 Seq Scan		3m 14s	88%
Hash Join	1	6s 634ms	3%
#4 Hash Join		6s 634ms	3%
HashAggregate	1	5s 798ms	3%
#3 HashAggregate		5s 798ms	3%
Foreign Scan	1	3,02ms	0%
#7 Foreign Scan		3,02ms	0%
Sort	1	0,102ms	0%
#1 Sort		0,102ms	0%
Hash	1	0,024ms	0%
#6 Hash		0,024ms	0%

Per index stats

Index	Count	Time
No index used		

Q1 / GGMD3 / Index - Lien Dalibo

Index utilisé : *nomprenom* de personne_insee.



Per table stats

Table	Count	Time	
public.personne_insee_large	1	1m 36s	84%
#6 Index Scan		1m 36s	84%

Per node type stats

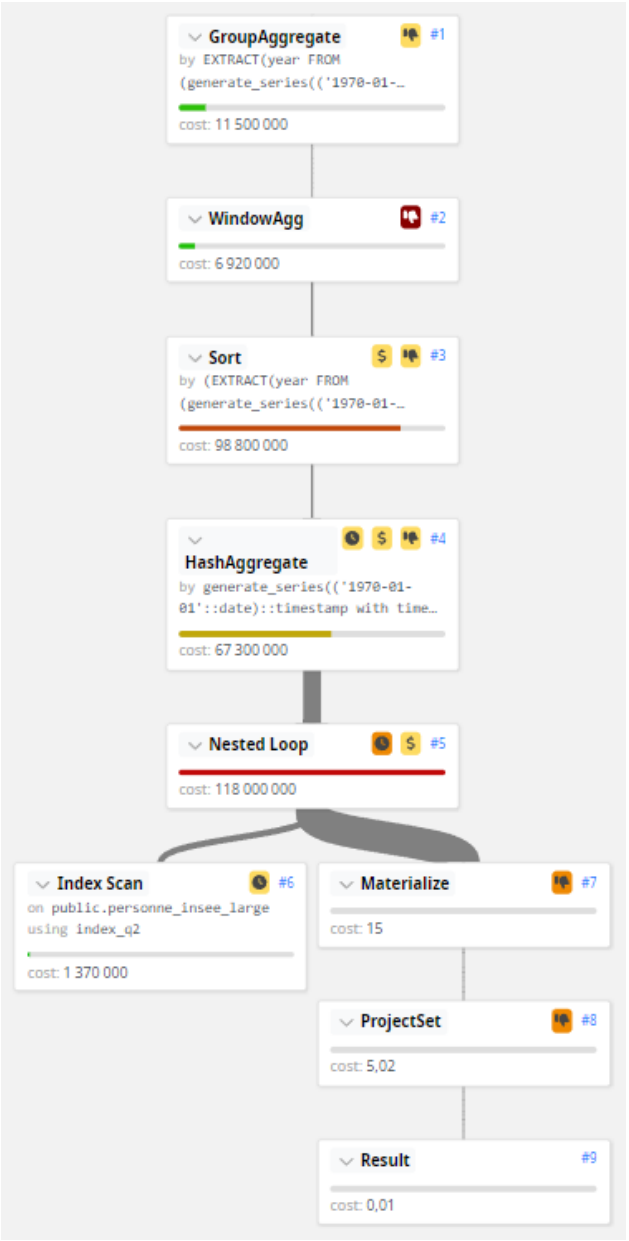
Node Type	Count	Time	
Index Scan	1	1m 36s	84%
#6 Index Scan		1m 36s	84%
Incremental Sort	1	9s 378ms	8%
#5 Incremental Sort		9s 378ms	8%
GroupAggregate	1	8s 260ms	7%
#4 GroupAggregate		8s 260ms	7%
HashAggregate	1	120ms	0%
#2 HashAggregate		120ms	0%
Subquery Scan	1	92,1ms	0%
#3 Subquery Scan		92,1ms	0%
Sort	1	0,121ms	0%
#1 Sort		0,121ms	0%

Per index stats

Index	Count	Time	
index_nomprenom	1	1m 36s	84%
#6 Index Scan		1m 36s	84%

Q2 / GGMD3 / Index - Lien Dalibo

Index utilisés : *datenaiss* et *datedeces* de personne_insee.



Per table stats

Table	Count	Time	
public.personne_insee_large	1	1m 23s	27%

Per node type stats

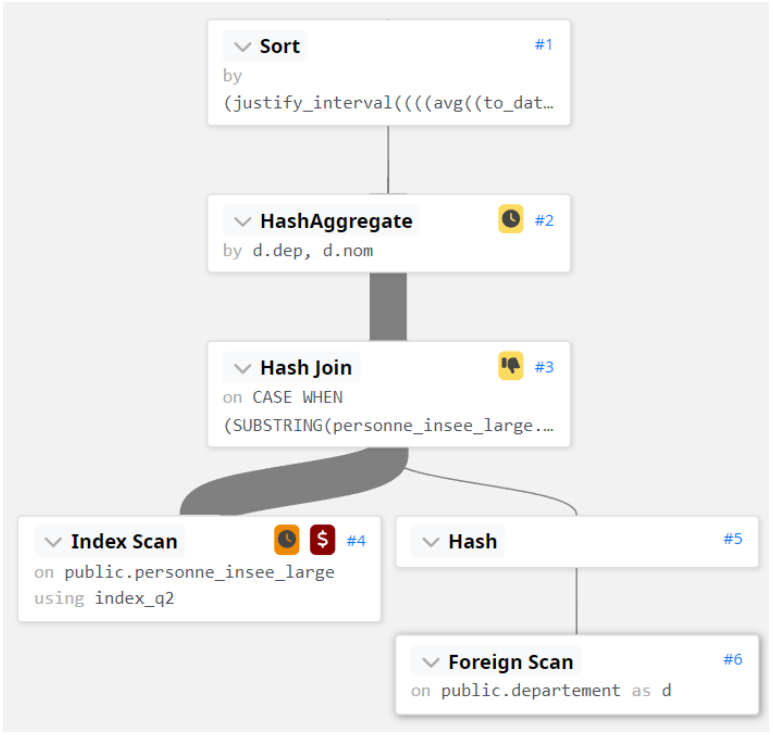
Node Type	Count	Time	
Nested Loop	1	3m 3s	59%
Index Scan	1	1m 23s	27%
HashAggregate	1	40s 465ms	13%
Sort	1	4s 834ms	2%
WindowAgg	1	493ms	0%
Result	1	248ms	0%
GroupAggregate	1	0,114ms	0%
ProjectSet	1	0,044ms	0%
Materialize	1	0ms	0%

Per index stats

Index	Count	Time	
index_q2	1	1m 23s	27%
#6 Index Scan		1m 23s	27%

Q3 / GGMD3 / Index - Lien Dalibo

Index utilisé : *datenaiss* et *datedeces* de personne_insee.



Per table stats

Table	Count	Time	
public.personne_insee_large	1	1m 18s	75%
public.departement	1	213ms	0%

Per node type stats

Node Type	Count	Time	
Index Scan	1	1m 18s	75%
HashAggregate	1	15s 889ms	15%
Hash Join	1	10s 571ms	10%
Foreign Scan	1	213ms	0%
Sort	1	0,051ms	0%
Hash	1	0,044ms	0%

Per index stats

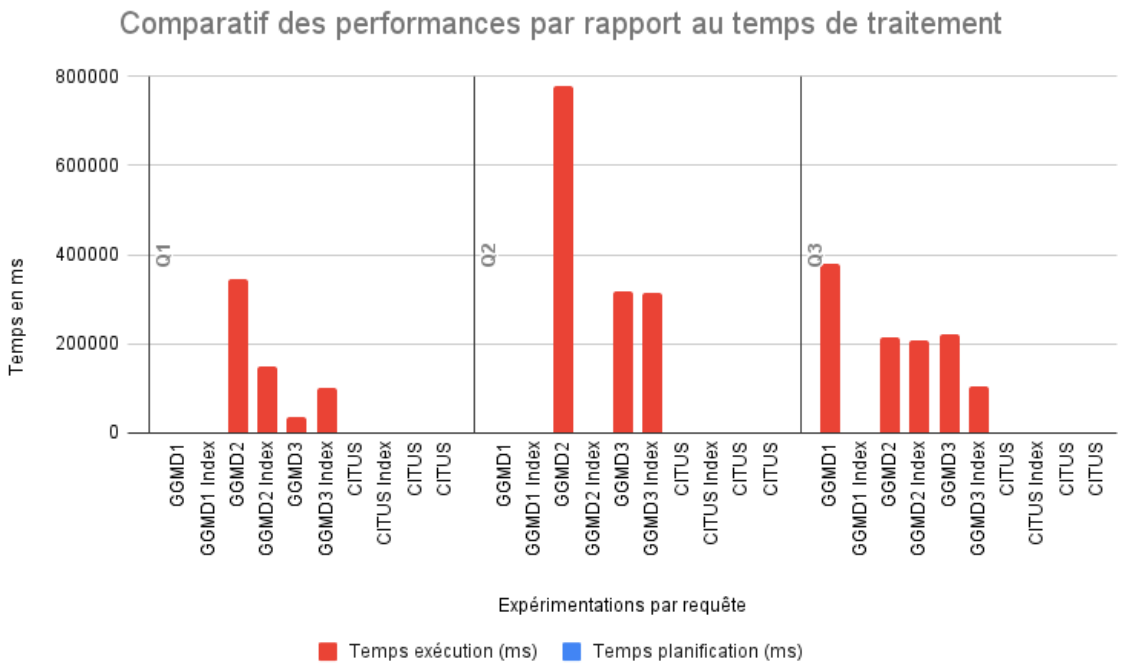
Index	Count	Time	
index_q2	1	1m 18s	75%

d. Sharding via Citus pour le traitement des requêtes

Cette partie n’a pas été traitée par manque de temps.

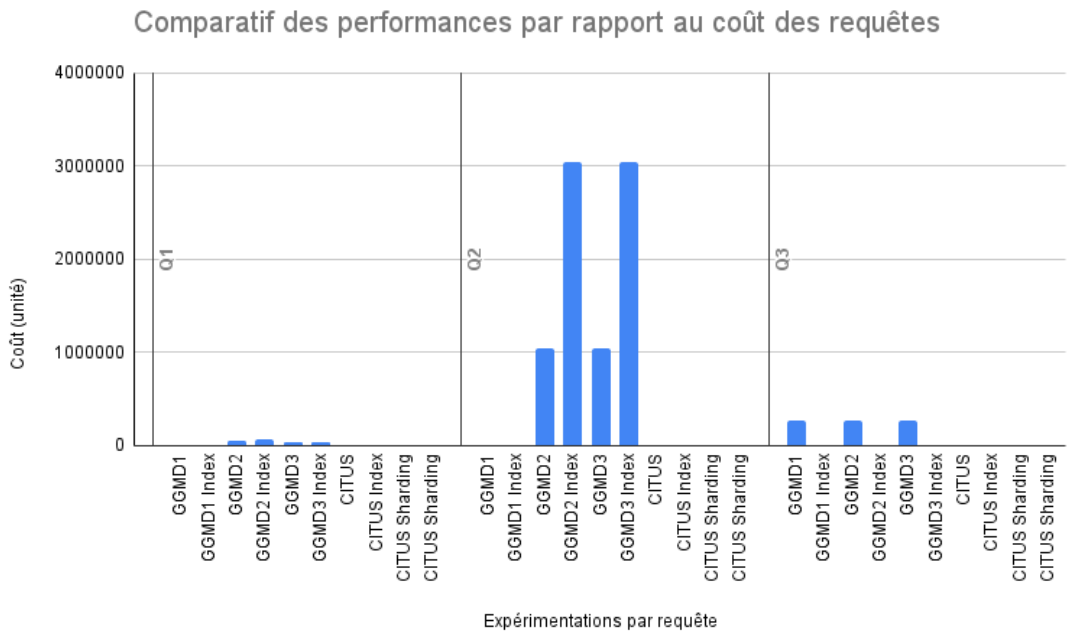
3. Graphes de synthèses

Graphe G1



Le temps de planification est totalement négligeable par rapport au temps d'exécution d'après nos relevés. De plus, on observe également une amélioration de la VM GGMD1 à la VM GGMD3 en temps de performance pour chaque requête.

Graphe G2



La requête Q2 est de loin la plus coûteuse des trois requêtes, peu importe sur quel VM elle est exécutée. On remarque aussi que le coût ne semble pas changer en fonction des VM mais plutôt en fonction des requêtes elles-mêmes.

4. Retour d’expérience

Pendant la réalisation du TP, nous avons constaté diverses variations de vitesse d'exécution des VM dans le temps. Le matin, lorsque le trafic sur l'infrastructure était relativement faible, les VM semblaient fonctionner à leur capacité maximale, exécutant rapidement les tâches qui leur étaient attribuées. En revanche, les après-midis et le soir, nous constations de forts ralentissements ce qui a eu un impact sur les résultats obtenus.

Il y avait aussi une disparité de vitesse d'exécution entre nos propres machines respectives lors de l'exécution des mêmes tâches sur une VM donnée. Par exemple, une requête qui s'exécute en seulement 10 minutes sur la VM d'une personne pouvait prendre plus d'une heure sur la VM d'un autre participant, alors même que ces VM étaient configurées de manière identique. Ce problème a duré quelques heures et certains jours mais nous a ralenti dans nos tâches respectives.

Pour la prochaine édition de ce TP, une requête ou une VM en moins allègerait grandement le TP et permettrait sûrement à plus de personnes de le terminer dans son intégralité.