
# RAPPORT TME 3 & 4 (SAM)

22 février 2024

Hichem BOUZOURINE (21319982)

# 1 DuckDB

Dans les TMEs 3 et 4, l'attention s'est portée sur l'optimisation des requêtes SQL, mettant en lumière les performances des différents systèmes de gestion de bases de données relationnelles, en l'occurrence DuckDB et SQLite. Le rapport suivant explorera les diverses stratégies mises en œuvre pour optimiser les requêtes, décrira les choix tactiques opérés et analysera le fonctionnement des principales optimisations déployées.

## 1.1 Exercice 1 Question a

Dans le cadre de l'optimisation des requêtes SQL, j'ai entrepris de traiter les sélections **avant les jointures** pour améliorer les performances de la requête donnée. Pour ce faire, j'ai utilisé des **requêtes imbriquées** afin d'appliquer les conditions de sélection avant d'effectuer les jointures.

Dans la requête initiale, les conditions de sélection étaient appliquées après les jointures, ce qui entraînait le traitement de toutes les lignes des tables jointes, y compris celles qui ne satisfaisaient pas les critères de sélection. Pour remédier à cela, j'ai utilisé des sous-requêtes pour filtrer les lignes des tables orders et lineitem avant d'effectuer les jointures avec la table customer.

Ma démarche a consisté à :

- Créer des sous-requêtes pour sélectionner uniquement les lignes de la table orders où la date de commande est antérieure au 15 mars 1995, et de la table lineitem où la date d'expédition est postérieure au 15 mars 1995.
- Utiliser ces sous-requêtes dans la clause FROM de la requête principale, en les associant à la table customer via des jointures JOIN basées sur les clés de liaison.
- Appliquer la condition de sélection c_mktsegment = 'BUILDING' dans la requête principale pour filtrer les clients appartenant au segment 'BUILDING'.
- Regrouper les résultats par priorité d'expédition (o_shippriority) et effectuer le comptage des occurrences pour chaque groupe.
- Trier les résultats par nombre d'occurrences (nb) dans l'ordre décroissant et limiter le nombre de résultats à 50.

J'ai ensuite exécuté la requête optimisée et vérifié les résultats.

En conclusion, cette approche a permis de traiter les sélections avant les jointures, ce qui a potentiellement amélioré les performances de la requête en réduisant le nombre de lignes à traiter lors des jointures. Ce rapport met en lumière les choix tactiques opérés et les étapes algorithmiques déployées pour optimiser efficacement la requête SQL.

## 1.2 Exercice 1 Question b

Ma démarche a consisté à restructurer la requête de manière à ce que les jointures soient effectuées dans un ordre différent de celui initial. Au lieu de joindre d'abord la table *customer* puis la table *lineitem*, j'ai choisi de joindre d'abord la table *orders* avec une *sous-requête* filtrant les clients du segment 'BUILDING', puis de joindre la table lineitem avec une autre sous-requête filtrant les éléments expédiés après le 15 mars 1995.

Plus précisément, j'ai :

— Modifié la clause FROM de la requête pour joindre d'abord la table orders, suivie des sous-requêtes filtrant les clients et les éléments expédiés conformément aux critères spécifiés.
— Utilisé des jointures JOIN pour associer les résultats de ces sous-requêtes à la table orders en fonction des clés de liaison.
— Appliqué la condition de sélection o_orderdate < CAST('1995-03-15' AS date) dans la requête principale pour filtrer les commandes passées avant le 15 mars 1995.
— Regroupé les résultats par priorité d'expédition (o_shippriority) et effectué le comptage des occurrences pour chaque groupe.
— Enfin, trié les résultats par nombre d'occurrences (nb) dans l'ordre décroissant et limité le nombre de résultats à 50.

Après avoir réécrit la requête, j'ai exécuté celle-ci et analysé le plan d'exécution ainsi que la durée de la requête pour évaluer l'impact de ce changement sur les performances.

En conclusion, cette approche a permis d'explorer l'effet de **l'ordre des jointures** sur les performances de la requête, en réorganisant les opérations de jointure pour optimiser l'accès aux données et potentiellement améliorer les performances globales. Le rapport met en évidence les étapes algorithmiques impliquées dans la réécriture de la requête et offre une analyse des résultats obtenus.