TD6 : Optimisation de requêtes

UCBL - Département Informatique de Lyon 1 – BDW - automne 2023

Objectif du TD : optimiser une requête algébrique, comparer des plans d'exécution

Ce jeu de données décrit des stations de vacances, leurs hôtels, les activités sportives ou culturelles que l'on peut y pratiquer, et les clientes qui ont fait des réservations dans ces hôtels. Les caractéristiques physiques des tables sont indiquées dans les tableaux ci-dessous :

- Les tailles données sont en octets
- La distribution des valeurs d'un attribut est uniforme dans son domaine
- Les valeurs d'attributs sont indépendantes (il n'y a pas de corrélation entre les attributs)

ACTIVITÉS (#*idS*, typeA, *description*, *nomClub*)
CLIENTES (idC, *nomC*, *prenomC*, *adresseC*, *telC*)
HOTELS (idH, *nomH*, *adresseH*, *catH*, *nbChambres*, #*idS*)
RÉSERVATIONS (#*idC*, #*idH*, dateR, *dureeR*)
STATIONS (idS, *nom*, *altitude*, *gare*, *région*)

Table	Nbre de n-uplets	Taille d'un n-uplet
STATIONS	1000	66
HOTELS	10000	122
ACTIVITÉS	10000	129
CLIENTES	25000	133
RÉSERVATIONS	20000	17

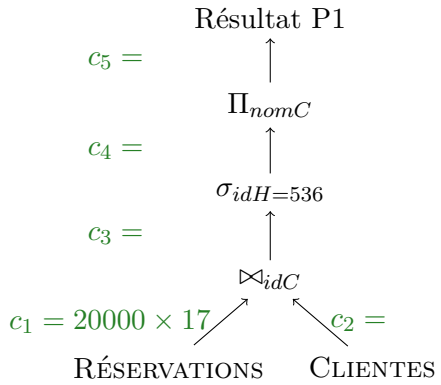
Tableau 1 : caractéristiques des tables

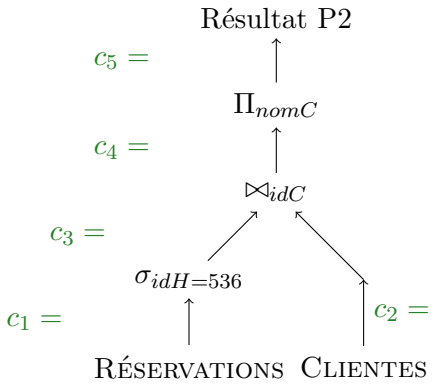
Attributs	Tables	Nombre de valeurs distinctes	Domaine	Taille
idS	STATIONS, HOTELS, ACTIVITÉS	1000	-	4
altitude	STATIONS	2000	$[0, 2000]$	2
idH	HOTELS, RÉSERVATIONS	10000	-	5
typeA	ACTIVITÉS	40	-	15
idC	CLIENTES, RÉSERVATIONS	25000	-	5
nomC	CLIENTES	25000	-	40

Tableau 2 : caractéristiques des attributs

Exercice 1 Calcul du coût d'exécution d'un plan

1. Traduire la requête suivante en SQL (avec un opérateur de jointure) : *le nom des clientes qui ont fait une réservation dans l'hôtel identifié par 536.*
2. Le SGBD a compilé la requête SQL précédente en deux plans algébriques (ci-dessous). Cette traduction vous semble-t-elle correcte ?





3. Pour chacun des deux plans algébriques, calculer le volume (en octets) de l'information manipulée pendant leur exécution, si celle-ci suit l'arbre syntaxique. Le coût c_1 du premier arbre (nombre de tuples $\times$ taille d'un tuple) est donné en exemple.
4. Les deux plans étant équivalents (résultat identique), que peut-on en conclure ?

Exercice 2 Optimisation algébrique

Nous nous intéressons maintenant au coût du plan d'exécution, en particulier pour la jointure lorsqu'elle est écrite sur le disque. Dans cet exercice, les hypothèses suivantes s'appliquent : l'opération de jointure est effectuée avec l'algorithme de boucles imbriquées avec bloc (*"block nested loops"*), la taille d'un bloc disque est de 1000 octets, la mémoire peut contenir 100 blocs[1], et le système peut lire 10000 blocs par seconde.

1. Calculez le coût de la jointure en secondes, dans le cas où cette opération est la première du plan d'exécution. Pour rappel, la formule du coût d'une jointure par boucles imbriquées par bloc est : $\frac{k_R}{m-1} \times (m - 1 + k_S)$ où m est le nombre de blocs en mémoire, k_R le nombre de blocs pour le premier opérande de la jointure, k_S pour le second opérande, avec $k_R < k_S$.
2. Pour réduire la taille des relations manipulées dans un arbre d'opérateur, il faut traiter les sélections avant les jointures (et produits cartésiens) et réduire le nombre d'attributs manipulés par des projections. Quelles sont les transformations algébriques à effectuer pour appliquer cette intuition ?
3. Choisissez un des plans de l'exercice précédent et optimisez le.
4. Calculez le nouveau coût de la jointure en secondes (après projection et sélection). La sélection s'effectue par itération (*"scan"*) de l'ensemble de la table. Que constatez-vous ?

[1] Cette valeur de 100 blocs pour la mémoire est plutôt faible.