Oracle SQL CheatSheet

Plan d'exécution

- Outil d'Oracle analysant comment une requête est exécutée ;
- Permet de mesurer les coûts en termes de performance ;
- Dans PL/SQL developer : File - New - ExplainPlan

Requêtes complexes

- le langage SQL n'est pas un langage procédural ;
- Éviter si possible les requêtes complexes qui répondent à plusieurs objectifs à la fois ;
  Écrire une requête pour chaque résultat recherché.

Résultats intermédiaires

- Penser à stocker un résultat intermédiaire s'il doit être utilisé plusieurs fois ;
- Décomposer les requêtes longues et complexes en petites requêtes ;
- Utiliser des vues matérialisées : grande quantité de données provenant de requêtes complexes.

Vues (non matérialisées)

- Éviter d'effectuer des jointures avec des vues complexes ;
- Éviter d'utiliser des vues créées pour des besoins spécifiques à d'autres circonstances ;
- Faire attention aux jointures externes sur les vues.

Requêtes SQL simples

Faire !

- Identifier les colonnes utilisant des fonctions par un nom significatif ;
- Donner un alias significatif aux tables et vues ;
- Mieux exploiter les DECODE et CASE
  minimiser la possibilité au PARSER d'accéder plus d'une fois à la même table

Ne pas faire !

- Éviter le * dans une requête SELECT
  sélectionner uniquement les colonnes nécessaires.
- Éviter de mettre des fonctions sur des colonnes indexées ;
- Éviter les LIKE
  utiliser le =
- Éviter les tris dans les requêtes imbriquées

IN - NOT IN - EXISTS - NOT EXISTS

- Privilégier EXISTS par rapport à IN (performance) ;
- Privilégier MINUS au lieu de NOT IN et NOT EXISTS (efficacité) ;
- Si possible : transformer les requêtes utilisant NOT EXISTS à l'aide de jointures externes (performance).

DISTINCT - UNION - ORDER BY

- Ces opérations engendrent des tris pouvant dégrader les performances ;
- Juger de leur nécessité avant de les utiliser ;
- Dans certains cas, on évite les tris en influençant les méthodes d'accès ;
- UNION ALL au lieu de UNION si possible.

FROM et WHERE

- Ordre des tables dans la clause FROM n'a pas d'importance ;
- Ordre des conditions dans le WHERE n'a pas d'importance
  Oracle optimise en appliquant les clauses les plus restrictives en premier
- Porter attention à bien imbriquer les AND et OR dans le WHERE.

Jointures

- L'index sur une colonne est ignorée quand une fonction y est appliquée ;
- L'ordre des jointures peut influencer les performances ;
- Éviter les anti-jointures autant que possible (>, <, ...) ;
- Utiliser l'équi-jointure = et le AND ;
- Éviter le mélange des types de données dans les conditions (conversions implicites) ;
- Exploiter les jointures lorsque la requête doit exploiter plusieurs colonnes
  Si la requête ne retourne qu'un résultat, faire une colonne avec un (select) au lieu d'une jointure
- Au lieu de faire les jointures dans le WHERE, on peut exploiter les clauses INNER JOIN, OUTER JOIN
  Oracle optimise en appliquant les clauses les plus restrictives en premier
- Porter attention à bien imbriquer les AND et OR dans le WHERE.

PL\SQL : scripts, procédures et packages

Séquences et INSERT INTO

S'assurer que la valeur courante (*currval*) de la séquence soit identique à la valeur MAX() du champ Séquence de la table où vous faites l'insertion.
Si vous n'avez pas les privilèges pour modifier la Séquence, dans une boucle, faites un 'séquence.NEXTVAL' jusqu'à atteindre la valeur MAX de la séquence de la table.

INSERT INTO ... RETURNING ... INTO

Si vous avez besoin d'une valeur générée lors d'une insertion, alors faites suivre la commande INSERT INTO d'un RETURNING {nom de la colonne} INTO {uneVariable}

Jointures et tris

- nested loops
- sort merge
- cluster
- hash

nested loops

Dans Oracle, il est possible d'ajouter des étiquettes aux boucles. Ajouter une étiquette à une boucle imbriquée permet d'expliciter une condition de sortie (EXIT) à celle-ci.

sort merge

Trier deux tables joints selon la clé de jonction. "Les jointures SORT-MERGE peuvent être efficaces lorsque le manque de sélectivité des données ou d'index utiles rendent une jointure NESTED LOOPS inefficace ou lorsque les deux sources de lignes sont assez volumineuses (plus de 5% des blocs utilisés)."

cluster

hash

Références

Formation RAMQ préparée par Momentum Technologies
dernière compilation : (5 octobre 2019)