**Title: SQL Database Performance Tuning**

**Author: Sravya**

**Track Category: Design**

**Subcategory: SQL script**

**Tags: SQL**

**Read Time: 5-8 min**

**Banner Image: https://lh3.googleusercontent.com/QGZ51tqWcbZCVtR1aoEZF-Ruc49RWB0CxvSX3GxI04Ef2tJ5ji8f7U86\_0rBlAMN7V4C=s152**

**Introduction: Performance Tuning steps for SQL Database**

To tune the performance of your SQL database, you should avoid unnecessarily loops in your SQL code

SQL database performance tuning is the process of ensuring that the SQL statements performed by an application run in the fastest route.

In medium and bigger companies, SQL database tuning usually handled by a Database Administrator (DBA), and many a time, developers also do the job of performance tuning in SQL Servers.

There are several ways to tune the performance of an SQL Server. Here, we will share a few tips and tricks to help you optimize the process of performance tuning in SQL Server.

1. Give prominence stage 1 over stage 2 predicates

2. Take out any/all scalar functions any/all mathematics coded on columns in predicates

3. Code only required columns needed in the select portion of the SQL statement

4. Stay away from distinct if possible

5. Always make sure the host variables are defined to match the column data types

6. Make sure that data distribution and other statistics are excellent and current in the tables processed

7. Use union all in steady of association where possible

8. Consider hardcoding versus using a host variable

9.Minimize db2's SQL requests

10. Try rewriting range predicates as between predicates

11. Consider using global temporary tables gtt

12. Remember that the order of using words equal range in the list like predicate

13.Implement good index design and specify the leading index columns in where clauses

14.Begin all filtering logic outside application code

15. Make use of dynamic SQL statement caching

16. Avoid using select \*

17. Watch out for nullable or times columns when SQL statements could have nulls returned from the database manager

18.Reduce the number of times open and close cursors

19. Avoid not logic in SQL

20. Keep table and index files healthy and organized

21.Know more about locking isolation levels

22. Take advantage of materialized query tables to improve response time ( dynamic SQL only )

23. Try using an insert with select

24. Take advantage of reopt once and reopt auto in dynamic SQL and reopt vars and reopt always in static SQL

**Dynamic SQL**

once - only one time in cache

auto - when values change

**Static SQL**

vars - same

always - same

25.Set your clustering index correctly

26. Watch out for tablespace scans

27. Use left outer joins over right outer joins

28. Take advantage of db2 v8 enhanced discard capabilities when it comes to mass deletes

29. Take advantage of the db2 load utility for mass inserts

30. Consider compressing data

31. Test your queries with realistic and a level of data to reflect performance issues

32. Use where instead of having for filtering whenever possible

33. Keep in mind index only processing whenever possible

34.Index on expression in db2 v9

35. Avoid sorts with an order by

36. Use joins instead of subqueries whenever possible

37.Know about skip locked data v9 for lock avoidance

38.Sort your input streams

39. Try index includes v10

40. Try using optimize for 1 row. It provides optimizer that the SQL intent may be to retrieve only a small amount of the whole result set and to give higher priority to the retrieval of the first row.