When you notice a significant slowdown in an SQL query's performance, there are several steps you can take to optimize it in Oracle 19c. Here’s a systematic approach:

1. Check Execution Plan

- Use `EXPLAIN PLAN FOR your\_query;` to analyze the execution plan. Look for any full table scans, large sorts, or other inefficient operations.

2. Statistics Gathering

- Ensure that your tables and indexes have up-to-date statistics. Use:

EXEC DBMS\_STATS.GATHER\_TABLE\_STATS('schema\_name', 'table\_name');

- For specific indexes:

EXEC DBMS\_STATS.GATHER\_INDEX\_STATS('schema\_name', 'index\_name');

3. Index Optimization

- Check if indexes are being used effectively. If not, consider adding indexes to columns used in WHERE clauses, JOIN conditions, or ORDER BY clauses.

4. Rewrite the Query

- Look for ways to simplify or restructure the query. Use Common Table Expressions (CTEs), subqueries, or JOINs more efficiently.

5. Partitioning

- If the table is large, consider partitioning it to improve query performance.

6. Database Configuration

- Review your database configuration settings (memory allocation, parallel processing, etc.) to ensure they’re optimal for your workload.

7. Monitoring and Resource Usage

- Use Oracle's built-in monitoring tools (like AWR reports or SQL Monitoring) to identify resource bottlenecks (CPU, I/O, etc.) during query execution.

8. Session Parameters

- Check for any changes in session parameters or the environment that might affect performance, such as `optimizer\_mode` or `pga\_aggregate\_target`.

9. Temporary Tables and Caching

- If the query can benefit from temporary tables or caching intermediate results, consider implementing those.

10. Database Load

- Investigate if there are other heavy processes running concurrently that could impact performance.

11. Review Recent Changes

- Consider any recent changes to the database schema, data volume, or application logic that might affect the query.

12. Testing and Iteration

- After making changes, test the query performance again. Sometimes, small adjustments can lead to significant improvements.

By following these steps, you should be able to diagnose and improve the performance of your SQL query. If the problem persists, consider consulting with a DBA for a more in-depth analysis.