

SQL PERFORMANCE: GUIDELINE FOR TUNING THE SQL QUERIES IN ORACLE RDBMS

A dissertation submitted to the Graduate School in partial
fulfillment of the requirements for the degree
Master of Science (Information Technology),
Universiti Utara Malaysia

By
Nor'azah binti Md Khushairi

Copyright © Nor'azah binti Md Khushairi , 2005. All rights reserved



JABATAN HAL EHWAL AKADEMIK
(Department of Academic Affairs)
Universiti Utara Malaysia

PERAKUAN KERJA KERTAS PROJEK
(Certificate of Project Paper)

Saya, yang bertandatangan, memperakukan bahawa
(I, the undersigned, certify that)

NOR 'AZAH BT. MD. KHUSHAIRI

calon untuk Ijazah
(candidate for the degree of) **MSc. (Information Technology)**

telah mengemukakan kertas projek yang bertajuk
(has presented his/her project paper of the following title)

**SQL PERFORMANCE: GUIDELINE FOR TUNING THE SQL QUERY
STATEMENTS IN ORACLE RDBMS**

seperti yang tercatat di muka surat tajuk dan kulit kertas projek
(as it appears on the title page and front cover of project paper)

bahawa kertas projek tersebut boleh diterima dari segi bentuk serta kandungan
dan meliputi bidang ilmu dengan memuaskan.
*(that the project paper acceptable in form and content, and that a satisfactory
knowledge of the filed is covered by the project paper).*

Nama Penyelia Utama
(Name of Main Supervisor): **ASSOC. PROF. NAZIB NORDIN**

Tandatangan
(Signature) : 

Tarikh
(Date) : 12 April 2005

PERMISSION TO USE

In presenting this thesis in partial fulfilment of the requirement for a postgraduate degree from Universiti Utara Malaysia, I agree that the University Library may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by my supervisor, in their absence, by the Dean of the Graduate School. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of material in this thesis, in whole or in part should be addressed to:

**Dean of Graduate School
Universiti Utara Malaysia
06010 UUM Sintok
Kedah Darul Aman.**

ABSTRAK

Sistem Pengurusan Pangkalan Data (SPPD) ialah sistem yang kompleks, di mana ia mengandungi himpunan sistem-sistem dengan tugas-tugas yang spesifik dan akan berinteraksi dengan pengguna aplikasi dan pangkalan data. Terdapat pelbagai aktiviti yang akan menyebabkan kemerosotan prestasi pangkalan data dan apabila ia berlaku, adalah amat penting untuk mendiagnosa, menganalisa dan menyelesaikan masalah ini secepat mungkin. Penyelidikan ini akan memfokus kepada kemerosotan prestasi pangkalan data yang disebabkan oleh 'SQL Query'. Kebanyakan pengaturcara aplikasi dan DBA menghadapi masalah untuk memahami langkah-langkah yang sepatutnya diambil untuk mempercepatkan pemrosesan 'SQL Query'. Garis panduan ini akan mengenalpasti factor-faktor yang menyebabkan kemerosotan prestasi pangkalan data dan 'SQL' yang bermasalah dan seterusnya menganalisa 'SQL' tersebut. Berdasarkan penyelidikan dan rujukan teknikal, satu metodologi untuk pembangunan garis panduan untuk mempercepatkan 'SQL Query' telah diperkenalkan. Item dalam metodologi tersebut termasuklah mendapatkan maklumat, menyediakan persekitaran untuk kajian, merancang kes kajian, mengenalpasti 'SQL' bermasalah, menggunakan alat untuk mempercepatkan 'SQL' bagi menganalisa, proses pengkodan dan proses analisa. Keputusan daripada analisa dan kajian telah dikompil, disusun dan seterusnya dijadikan garis panduan untuk mempercepatkan 'SQL Query'. Garis panduan yang dihasilkan dikategorikan kepada tiga bahagian. Bahagian yang pertama menerangkan tentang skrip untuk mengenalpasti SQL yang bermasalah. Bahagian yang kedua mengenai 'SQL Counter' iaitu program yang telah dibangunkan bertujuan untuk mengenalpasti 'SQL' yang bermasalah di dalam 'nested procedure'. Apabila 'SQL' telah dikenalpasti, alat untuk menganalisa 'SQL' dengan tujuan mencari masalah 'SQL' tersebut dan mempercepatkannya digunakan. Eksperimen telah dijalankan untuk menghasilkan garis panduan ini dan juga untuk membuktikan ia dapat mencapai objektif yang telah ditetapkan. Kesimpulannya, garis panduan ini dapat menjadi praktis yang bagus dan amat berguna untuk pengaturcara SQL atau pun DBA.

ABSTRACT

DBMS is a complex system that is composed of a collection of subsystems, each with a specific task and will interact with the user application programs and the database. Maintaining DBMS performance is a challenging task. Many events can cause the database poor performance and once it occurs, it is important to diagnose and resolve the problem quickly. The research will focus on poor performance caused by SQL Query. Most of application developer and DBA have problem to understand the SQL Query tuning step. This guideline will determine the cause of database performance problem, identify the bad SQL and analyze it. Based on the literature review, methodologies have been introduced which cover all the steps needed to come out with a guideline. The methodology stages are gathering information, set up environment, design test cases, identify problematic SQL, use tuning tool to analyze the SQL, coding process and analysis. The results from the analysis and experiment have been compiled and arranged to get the SQL Query Tuning Guideline. This guideline has been separated into 3 sections. Firstly, explains on the scripts to identify the problematic SQL. Secondly, on the SQL Counter, the program that has been developed in order to spot poor performance SQL in the nested procedure. Finally, using tuning tool, Explain Plan and TKPROF to analyze the SQL statement. Experiments have been conducted in order to output this guideline and to prove that this guideline is able to help the user to tune the SQL. Furthermore, this guideline also introduces a new technique, SQL Counter, in identifying the problematic SQL in the nested program or sub query. This guideline can be a good practice to the developers or the DBAs themselves and can be regard as one useful recommendation.

ACKNOWLEDGEMENTS

I would like to express my gratitude to all, in one way or another, contributed to the work of the thesis and for the completion of my Master Degree study.

First and foremost, I would like to thank my supervisor, Assoc. Prof Nazib Nordin who has been extremely supportive and patience in dealing with me. My journey has not always been easy, but he has been with me every step of the way. He had given encouragement during my ups and downs especially at the time when I lost my beloved father. At that time, I feel so weak, disappointed and down until I feel that I could not carry on with this research. His advice and motivation makes me realize that life needs to go on.

Heartfelt thanks to my colleague at work, Mohd Suhaimi, Annusia, Zaidah Hanim, Shakirah who had given their views and comments to me until the completion of this research.

I would like to express my truthful thanks to my beloved parents, for their prayers and constant support. A very special thanks to my father, Allahyarham Md Khushairi Budin. He is my idol ever after. I would not forget his courage to see me as a person that I am now. Not forgetting, my mother, Aishah Isa, who always been caring and supportive.

Utmost thanks to my beloved husband, Mohamed Rodzi, for giving me the chance to further my studies. Thanks for the understanding, patience and support because without this I could never complete my studies.

Last but not least, my gratitude to God for giving me the courage to persevere with high dedication till the accomplishment of my Masters degree study.

TABLE OF CONTENTS

PERMISSION TO USE	I
ABSTRAK	II
ABSTRACT	III
ACKNOWLEDGEMENTS	IV
TABLE OF CONTENTS	V
LIST OF TABLE	X
LIST OF FIGURES	XI
CHAPTER 1 : INTRODUCTION	1
1.1 Overview	1
1.2 Definition.....	2
1.3 Problem Statement.....	4
1.4 Project Objectives.....	5
1.5 Significance Of The Research	5
1.6 Scope And Limitation.....	6
1.7 Report Structure.....	6
CHAPTER 2 : LITERATURE REVIEW	8
2.1 Related Works on Database Optimization.....	9
2.2 Related works on DBMS Workload	11
2.3 DBMS Tuning	12
2.4 Related works on SQL Tuning	13
2.6 SQL Guideline	17
2.7 SQL Query Tuning Tool.....	18
2.7.1 Automatic SQL Tuning	18
2.7.2 LECCO DB Expert	19

2.7.3 Oracle Expert.....	19
2.7.4 Explain Plan.....	20
2.8 Conclusion	20
CHAPTER 3 : RESEARCH METHODOLOGY	21
3.1 Introduction	21
3.2 Components Of The Experiment Method	22
3.2.1 Participants	23
3.2.2 Elements & Factors	23
3.2.3 Instruments and materials used in this experiment.....	23
3.2.4 Procedures and measures query effectiveness.....	24
3.2.5 Statistical analysis.....	24
3.2.6 Experimenter.	24
3.3 Research Methodology Stages Overview.....	25
3.4 Gathering Information	26
3.5 Setup Environment	26
3.5.1 Set Up Environment	26
3.5.2 DBMS Assumptions.....	27
3.6 Design Test Cases.....	28
3.6.1 Identify high impact SQL queries.	28
3.6.2 SQL Queries Illustration.....	29
3.7 Identify Problematic SQL Query.....	29
3.7.1 SQL Script	30
3.7.2 Develop SQL Loop Counter.....	30
3.8 Using Tool to Analyze the SQL	31
3.8.1.EXPLAIN PLAN.....	31
3.8.2 TKPROFF.....	31
3.9 Analysis	32
3.10 Document the Guideline.....	32
3.11 Conclusion	32

CHAPTER 4 : EXPERIMENTS AND RESULTS	33
4.0 Introduction	33
4.1 Experiment.....	34
4.2 Scripts to Identify Problematic SQL	36
4.2.1 Script to identify the resource-intensive code	36
4.2.1.1 Parameter	36
4.2.1.2 The Script	37
4.2.1.3 Sample Output	39
4.2.1.4 Tips and Technique	40
4.2.2 Script to Get the SQL Text.....	41
4.2.2.1 Parameter	41
4.2.2.2 The Script	41
4.2.2.3 Sample Output	42
4.2.3 Script to get the worst SQL running.....	43
4.2.3.1 Parameter	43
4.2.3.2 The Script	43
4.2.3.3 Sample Output	46
4.2.4 Script to identify users currently running the SQL.....	47
4.2.4.1 Parameter	47
4.2.4.2 The Script	47
4.2.4.3 Sample Output	47
4.2.5 Analyzing SQL Execution: Script to Count Cartesian Join .	48
4.2.5.1 Parameter	48
4.2.5.2 The Script	48
4.2.5.3 Sample Output	49
4.2.6 Analyzing SQL Execution: Script To View Cartesian Join SQL Statement.....	50
4.2.6.1 Parameter	50
4.2.6.2 The Script	50
4.2.7 Analyzing SQL Execution: Script to Check for SQL on Table	

Scan	50
4.2.7.1 Parameter	50
4.2.7.2 The Script	51
4.2.7.3 Sample Output	52
4.2.8 Analyzing SQL Execution: Statistic Script	53
4.2.8.1 Parameter	53
4.2.8.2 The Script	53
4.2.9 Scripts Summary.....	54
4.3 Scripts to Spot Poorly Performance SQL in Nested Programs	55
4.3.1 Introduction	55
4.3.1.1 Nested Procedures or Functions	56
4.3.1.2 Unknown SQL.....	57
4.3.2 Introduction to SQL Counter.....	58
4.3.2.1 What is SQL Counter?.....	58
4.3.2.2 How does SQL Counter work?.....	58
4.3.3 SQL Counter Flow Diagrams.....	59
4.3.4 What SQL Counter Can Do?.....	60
4.3.4.1 Spot ineffective SQL statements down to the lowest level recursively.....	60
4.3.4.2 Identify poorly written SQL codes from remote application.	61
4.3.4.3 Capture unlimited nonproductive SQL codes from a particular session	62
4.3.4.4 Database server performance Monitoring	63
4.3.5 Steps to use SQL Counter.....	64
4.4 Using Tuning Tool to Analyze the SQL statement	66
4.4.1 Execution Plan.....	66
4.4.2 Explain Plan.....	66
4.4.2.1 Explain Plan Characteristic	66
4.4.2.2 Understanding Explain Plan	67
4.4.2.3 Explain Plan Table.....	68

4.4.2.4 Sample of simple Explain Plan Output.....	69
4.4.3 TKPROF	69
4.4.4 Explain Plan Tips.....	70
4.4.5 Steps to generate EXPLAIN PLAN.	71
4.4.5.1 Check plan_table existence	71
4.4.5.2 Run EXPLAIN PLAN	71
4.4.5.3 Format the plan table	71
4.4.5.4 Displaying the execution plan.	72
4.4.6 Steps to generate TKPROF Reports	73
4.4.6.1 Enabling SQL TRACE	74
4.4.6.2 Timed Statistics	74
4.4.6.3 Finding the SQL trace file	75
4.4.6.4 Using TKPPROF	75
4.4.6.5 TKPPROF Reports	76
4.5 Conclusion	77
CHAPTER 5 : RECOMMENDATIONS AND CONCLUSION.....	78
5.1 Introduction	78
5.2 Problem Synopsis	79
5.3 Evaluation.....	80
5.4 Obstacles and Limitations	80
5.5 Recommendations for Future Project.....	81
5.6 Conclusion.....	82
REFERENCES	83
APPENDICES	87

LIST OF TABLES

Table 4.2.9: Identifying Problematic SQL	54
Table 4.4.2.3 : Important Field Plan_Table	68

LIST OF FIGURE

Figure 3.2 Experimenter Method Plan	22
Figure 3.3 Proposed Research Methodology	25
Figure 4.1 : Steps to identify and test the tuning scripts	35
Figure 4.2.1.3 : Sample Output to identify the resource-intensive code	39
Figure 4.2.2.3: Sample Output for scripts to get the SQL text	42
Figure 4.2.3.3: Sample Output of SQL to get the worst SQL running	46
Figure 4.2.4.3 Sample Output, Identify user	47
Figure 4.2.5.3 : Sample Output, Count Cartesian Join	49
Figure 4.2.7.3: Sample Output, SQL on table scan	52
Figure 4.3.1.1 Nested Procedures or Functions	56
Figure 4.3.1.2 Unknown SQL	57
Figure 4.3.3 SQL Counter Flow Diagrams	59
Figure 4.3.4.1 Nested Procedures and Functions in DBMS	60
Figure 4.3.4.2 Application running in DBMS environment	61
Figure 4.3.4.3 SQL codes from a particular session	62
Figure 4.3.4.4 SQL Counter Output Log vs System Performance	63
Figure 4.3.5 (i) : Result when viewing active load in the database	64
Figure 4.3.5(ii): Command to run the script.	64
Figure 4.3.5(iii) : List of log file available.	65
Figure 4.4.2.4 Simple Explain Plan	69
Figure 4.4.5.4 Sample Output of Execution Plan	73
Figure 4.4.6.5 Sample TKPROF Output Report	76

CHAPTER 1

INTRODUCTION

1.1 Overview

Achieving a high level performance from a Database Management System (DBMS) has been a difficult task. Maintaining data integrity and supporting concurrent users introduce a significant amount of overhead to a DBMS. This overhead affects the ability of the DBMS to serve the data to the users quickly. As such, decreasing the overhead of DBMS while maintaining data integrity and providing information to the user as quickly as possible has been the main objective of database operation in organizations. It involves determining which of the numerous resources need to be adjusted in order to solve the performance problems. Performance issues are often not identified until the developer has done the coding, either in the integration testing, Quality Assurance (QA) or worst case, at run time (LeccoTechnology, 2002). Currently, developers focus mainly on how to code a SQL statement to retrieve the correct set of data from the database.

The contents of
the thesis is for
internal user
only

REFERENCES

- Cambridge (2003). Cambridge Advanced Learner's Dictionary. Cambridge International Press : Melbourne, Australia. Cambridge.
- Connolly, T.M., & Begg, C. E. (2002). SQL: Data Manipulation in A.D. McGettrick (3rd Ed). Database Systems: A Practical Approach to Design, Implementation and Management. (pp 110-155). Harlow: Addison Wesley.
- Agrawal, S., Chaudhuri, S. & Narasayya, V. (2000). Automated Selection of Materialized Views and Indexes for SQL Databases, Proceedings of the 26th International Conference on Very Large Databases, September 10-14, Cairo, Egypt, pp. 496-505, Morgan Kaufmann Publishers.
- Talwadder, A.S. (2003). Survey of performance issues in parallel Database Systems. ACM Journal of Computing Sciences in Colleges, Consortium for Computing Sciences in Colleges, Volume 18 (6), 5-9
- Taniar, D., Tan, R.B, Leung, C.H.C & Liu, K.H. (2003). Performance analysis of "Groupby-After-Join" query processing in parallel database systems. Journal of Information Sciences, Volume 168 (1-4), 25-50
- Graefe, G. (1993). Query Evaluation Techniques for Large Databases. ACM Computing Surveys (CSUR). Volume 25(2), 73-169.
- Chaudhuri, S., Narasayya, V., Ramamurthy, R. (1999). Research sessions: query progress: Estimating progress of execution for SQL queries, International Conference on Management of Data, Proceedings of the 1999 ACM SIGMOD international conference on Management of data, 803-814
- Rao, J., Pirahesh, H., Zuzarte, C. (2004). Canonical Abstraction for Outerjoin Optimization. International Conference on Management of Data, Proceedings of the 2004 ACM SIGMOD international conference on Management of data, Paris, France, 671-682
- Sadri, R. (2004). Expressing and Optimization Sequence Queries in Database Systems, ACM Transactions on Database Systems (TODS), Volume 29(2), 282-318
- Luo, G., Naughton, J.F., Ellmann, C.J., Watzke, M. W. (2004). Toward a Progress Indicator for Database Queries. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. Paris, France. 791-802

- Chaudhuri, S. & Narasayya, V. (2000). Automating Statistics Management for Query Optimizers. Proceedings of 16th International Conference on Data Engineering, pp. 339-348, San Diego, USA.
- Oracle (2004). Oracle 9i Performance Tuning Manual: Overview of Oracle 9i Database Performance Tuning Manual. (pp 1-4). Oracle.
- Dageville, B., Das, D., Dias, K., Yagoub, K., Zaid, M. & Ziauddin, M. (2004). Automatic SQL Tuning in Oracle 10g. Proceedings of the 30th VLDB Conference, Toronto, Canada, 1098-1109
- Leccotech (2002a), SQL Performance Tuning and Optimization Solution. Retrieved July 17, 2004, from http://www.leccotech.com/pdf/DB2_DS.pdf
- Leccotech (2002b), Oracle Application Optimization and Performance, The SQL Problem. Retrieved March 12, 2005, from <http://oracle.ittoolbox.com/browse.asp?c=OraclePeerPublishing&r=%2Fpub%2FCB041802c%2Epdf>
- Larsen, S.M. (1999), Top Ten SQL Performance Tips, Retrieved July 9, 2004, from http://www.quest.com/whitepapers/10_SQL_Tips.pdf?Offer=dwpr0325
- Computer Associates (2002), Oracle SQL and PL/SQL Development and Maintenance Concepts, Retrieved July 10, 2004, from http://www3.ca.com/Files/WhitePapers/best_practices_white_paper.pdf
- Dinda, P. A. & Lu, D. (2003). Nondeterministic Queries in a Relational Grid Information Service. Retrieved Oct 1, 2004, from <http://www.cs.northwestern.edu/~urgis/NWU-CS-03-15.pdf>
- Mallette, D. (2003) Achieving and Maintaining Optimal Application Performance, Retrieved July 10, 2004, from http://www.tssug.com/presentations/Achieving_and_Maintaining_Optimal_Application_Performance_April_15_2003.pdf
- Haecke, B. (2002). JDBC 3.0-Java Database Connectivity. M&T Books, New York, Cleveland, Indianapolis. Retrieved July 17, 2004 from <http://www.netobjectdays.org/node02/de/Conf/publish/../../../../pdf/02/papers/ws-jada/1149.pdf>

- Beeler, B. & Rodriguez, I. (2002). Database Performance Optimization Made Easy. Retrieved Dec 14, 2004, from <http://www.naspa.com/PDF/2002/1202%20PDF/T0212005.pdf>
- Shrag, R.(2000).Use EXPLAIN PLAN and TKPROF to Tune Your Applications Retrieved Dec 14, 2004, from http://www.dbspecialists.com/presentations/use_explain.html
- Darmawan, B., Groenewald, G., Irving, A., Henrique, S., Monteiro, S. & Snedeker, K.M. (2003). Database Performance Tuning on AIX (pp 291). IBM.
- Trezzo, J. C. (1999). Oracle PL/SQL Tips & Technique. Osborne:MsGraw-Hill.
- Tu.S.W. (2000). From Guideline Modeling to Guideline Execution : Defining Guideline- Based Decision- Support Services. Retrieved Aug 20, 2004. from http://www-smi.stanford.edu/pubs/SMI_Reports/SMI-2000-0829.pdf
- IBM(2002). Administration Guide : Performance (version 8), Retrieved Aug 20, 2004. from : <http://users.sdsc.edu/~jrowley/db2/Admin%20Gd%20-%20Performance.pdf>
- Elmastri, R. & Navathe, S.B.(2004). Fundamental of Database System (4th Ed). Boston: Pearson.
- Gupta, A., Davis, K.C., Litton, J.G.(2002). Performance Comparison of Property Map and BitMap Indexing. Retrieved Aug 20,2004. from <http://www.cis.drexel.edu/faculty/song/dolap02/paper/p65-gupta.pdf>
- Elnaffar, S.S.(2002). A Methodology for Auto-recognizing DBMS Workloads. Retrieved March 12,2005. from <http://www.cs.queensu.ca/home/elnaffar/Publications/cascon2002.pdf>
- Aronoff, E.,Loney,K.,Sonawalla,N.(1997). Quest Software, Explain Plan : Everything you wanted to know and had no one to ask. Retrieved March 10,2005 from <http://www.lifl.fr/~bossut/Docu/explain.pdf>
- Vaidyanatha,G.K.(2002). Oracle Database Performance Management. Retrieved March 10, 2005 from <http://www.lifl.fr/~bossut/Docu/explain.pdf>

Rao,M.,Shah,S.(2002).Experimentation A Research Methodology.
Retrieved March 10,2002 from
[http://www.public.asu.edu/~kroel/www500/EXPERIMENTATION%20Fr
i.pdf](http://www.public.asu.edu/~kroel/www500/EXPERIMENTATION%20Fr
i.pdf)