Designing for Performance: Database Related Worst Practices

ITOUG Tech Day, 11 November 2016, Milano (I)

Christian Antognini

@ChrisAntognini

Senior principal consultant, trainer and partner at Trivadis in Zurich (CH)

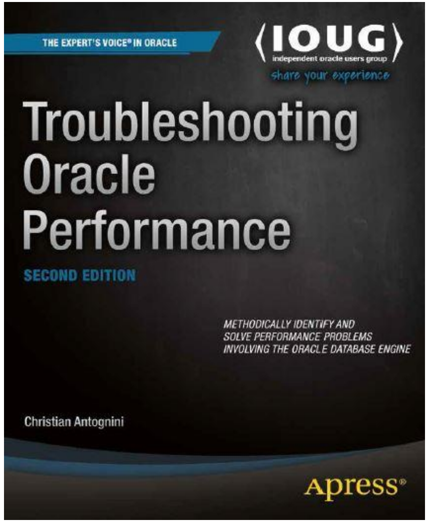
- christian.antognini@trivadis.com
- http://antognini.ch

Focus: get the most out of Oracle Database

- Logical and physical database design
- Query optimizer
- Application performance management

Author of Troubleshooting Oracle Performance (Apress, 2008/14)

OakTable Network, Oracle ACE Director



THE EXPERT'S VOICE® IN ORACLE

IOUG independent oracle users group

share your experience

Troubleshooting Oracle Performance

SECOND EDITION

METHODICALLY IDENTIFY AND SOLVE PERFORMANCE PROBLEMS INVOLVING THE ORACLE DATABASE ENGINE

Christian Antognini

Apress®

OakTable®

trivadis

makes IT easier.

Agenda

1. Introduction
2. Worst Practices
3. Core Messages

Introduction

Where Does This Information Come From?

Personal experience!

I spent most of the last 15 years troubleshooting performance problems.

- Hundreds of different projects and applications
- A number of industries were involved

According to my experience, most of the database design related performance problems are caused by a limited number of issues.

- In this presentation I cover the top 10
- Even though some of them may seem very basic, do not underestimate the number of systems that are now running and suffering because of them!

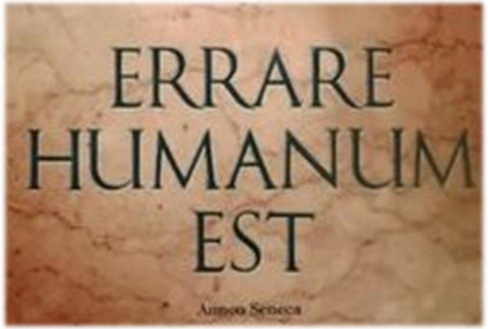
Why It’s Important?

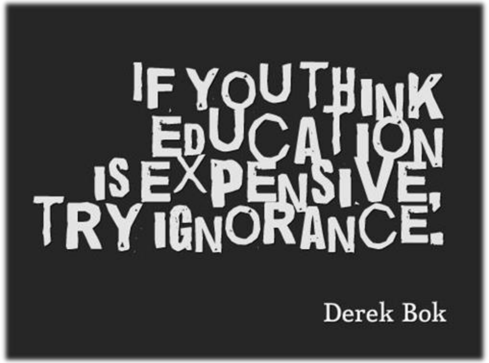
Many performance problems can be avoided with sound planning and design.

- Do the right thing!

There is a widespread lack of knowledge in our industry!

- The same mistakes are made again and again.
- “To err is human, to persist in it, is diabolical” – Seneca





I FEEL COMFORTABLE MAKING THE SAME MISTAKES OVER AND OVER AND OVER AGAIN. AT LEAST I KNOW WHAT I'M DOING.
berner

Source: @peter_berner

It’s Not Premature Optimization!

“Premature optimization is the root of all evil” – Donald Knuth

The misconception based on that particular quote is that a developer, initially, should ignore optimization altogether.

In my opinion, this interpretation is wrong!

My take:

- Developers should avoid micro optimizations that have local impact.
- Developers should care about optimizations that have global impact, like the design of a system, the algorithms used to implement the required functionality, or in which layer a specific processing should be performed.

Worst Practices

Lack of Logical Database Design

Once upon a time, it was considered obvious that one should have a data architect involved in every development project.

Today, unfortunately, too often I see projects in which no formal database design is done.

- The database is considered a dumb device.

The logical database design depends on the target database engine.

- Schema-on-write vs schema-on-read

Implementing Generic Tables

In the quest of *flexibility*, generic database designs are implemented.

- Entity-attribute-value models
- XML- and JSON-based designs

Flexibility is tied to performance!

- In some situations suboptimal performance might be good enough.
- In other situations it might be catastrophic.

Not Using Constraints to Enforce Data Integrity

Constraints are not only fundamental to guarantee data integrity, but they are also extensively used by query optimizers during the generation of execution plans.

Cons of checking the constraints at the application level:

- More code being written and tested
- Potential problems with data integrity
- Greater consumption of resources (not always)
- Leads to less scalable locking schemes (not always)

Lack of Physical Database Design

It is not uncommon to see projects where the logical design is directly mapped to the physical design.

The physical database design depends on target database engine.

- Heap tables, clustered tables, IOT
- Many type of indexes exist
- Data partitioning

Do not forget to define and implement a data-archiving concept.

Not Choosing the Right Data Type

In recent years, I have witnessed a disturbing trend in physical database design: *wrong datatype selection*.

- E.g. VARCHAR2 instead of DATE or TIMESTAMP

There are four main problems related to wrong datatype selection:

- Wrong or lacking validation of data
- Loss of information
- Things do not work as expected
- Query optimizer anomalies

Not Using Bind Variables Correctly

From a performance point of view, bind variables introduce both an advantage and a disadvantage.

- Only applies to database engines having a cursor cache.

The advantage of using bind variables is that they allow the sharing of cursors.

The disadvantage of using bind variables in WHERE clauses, and only in WHERE clauses, is that crucial information is sometimes hidden from the query optimizer.

From a security point of view, bind variables prevent the risks associated with SQL injection.

Not Using Advanced Database Features

High-end database engines provide many advanced features that can drastically reduce development costs while boosting performance.

- E.g. data-centric processing should take place as closely as possible to the data

Leverage your investment by taking advantage of those features as much as possible.

Most of the time, database-independent applications are not required.

- I would recommend to develop a database-independent application only when there are very good reasons for doing it.
- Companies are more likely to change the whole application before changing just the database engine.

trivadis
makes IT easier.

Performing Unnecessary Commits

Commits are operations that call for serialization.

Every operation that leads to serialization inhibits scalability.

Serialization is unwanted and should be minimized as much as possible.

Steadily Opening and Closing Database Connections

Opening a database connection that in turn starts an associated process or thread on the database server is *not* a lightweight operation.

- Do not underestimate the amount of time and resources required.
- A worst-case scenario that I sometimes observe is a web application that opens and closes a database connection for every request that involves a database access.

Using a pool of connections is of paramount importance.

Opening Too Many Database Connections

A database connection that has an associated process or thread on the database server is *not* a lightweight resource.

A processor core cannot efficiently handle many concurrent processes or threads.

Using a pool of connections of limited size is of paramount importance.

Core Messages

Let’s Face IT, Information Technology Is Expensive!

Simple solutions
for simple problems,
elegant solutions
for complex problems.

One Size Fits All?

Use the right tool
for the right job.

Performance Is Not an Option

Optimal performance is not simply a product one can buy but rather the results of an accurate planning and a correct implementation.

Questions and Answers

Christian Antognini

Senior Principal Consultant
christian.antognini@trivadis.com