SQL Server: Why Physical Database Design Matters

Module 1: Introduction

Kimberly L. Tripp
Kimberly@SQLskills.com
http://www.SQLskills.com/blogs/Kimberly

pluralsight
hardcore developer training

Does Physical Database Design Matter?

- **The database just deals with x or y or z, design doesn't matter…**
- **Let's just get it done**
  - "We'll deal with performance later…"
- **Disk space is cheap**
  - "Who cares about 4 bytes vs. 8 bytes vs. 16 bytes…"
- **Unfortunately, the sooner you start to write code, the longer it's going to take!**
- **The time spent doing a minimal amount of prototyping is really WELL WORTH IT!**
  - Helps you to build better, more scalable applications
  - Helps you to find out early in the development process:
    - Feature requirements
    - Feature combinations and their limitations
    - Performance problems

Why is This Course Important?

- **Some applications will work well… for a long time**
  - Small applications
  - Few users
  - Possibly critical applications but just not high volume
  - But, how often has an application been slated to be a short-term solution…
- **Truly scalable applications don't happen by accident**
  - There is no platform that's perfect for every possible use case
- **Can you successfully use a power tool without reading the manual?**
  - You might not get the full power of the tool
  - You might do something the hard way or end up breaking the tool
  - You might not be successful at all

What Does Physical Database Design Mean?

- **Two aspects to design: logical modeling and physical implementation**
- **Often people think that physical implementation is just a matter of knowing the syntax of that particular RDBMS**
  - PL/SQL: Procedural Language/SQL for Oracle
  - T-SQL: Transact-SQL for Microsoft SQL Server
  - SQL PL: SQL Procedural Language for IBM DB2
- **What this course will <u>prove</u> is that your design decisions should take the actual platform into consideration**
  - Understanding the platform can help you to design a more scalable, available, and reliable database
- **To create a truly scalable database takes 3 things:**
  - Know your data
  - Know your workload
  - Know how the platform works!

Prototyping Makes Perfect

- **Common scenario**
  - Unable to leverage online operations
- **Common cause**
  - Enterprise Edition of SQL Server 2008/2008 R2 supports online index maintenance operations, but there are restrictions that aren't lifted until SQL Server 2012
- **Possible Solution(s)**
  - Minimize fragmentation by changing the clustering key
    - Problem: requires downtime when the clustered index is the primary key
  - Use vertical partitioning to move the problematic columns to another table
    - Problem: all of the application code will need to change
- **Cost**
  - Availability – depending on table size
  - Development time – testing the new solution
- **Lessons learned**
  - Understanding the relationship between certain data types, index fragmentation, and maintenance could have helped to avoid this
  - Prototyping the design could have helped to avoid this

Why Is This Course Relevant

- **There are some very common scenarios that are not well understood**
- **I've watched arguments on websites, blog posts, in-person… often, based on assumptions and misunderstandings about what SQL Server seems like it is doing**
- **Microsoft SQL Server is a multi-purpose, relational, database engine**
  - It can do anything
  - It can house any data
  - But, that doesn't mean that the defaults for EVERYTHING are good for EVERY one and EVERY database
- **Most mistakes have relatively simple solutions that would have been trivial to implement early-on in the development process**
  - But once that application is in production, the changes might be anywhere from challenging to impossible (without great concessions such as downtime)

Why Can I Teach This Course?

- **I've been working with SQL Server since 1990 as a consultant, trainer, speaker, and author**
  - I've watched common patterns emerge
  - I've studied them
  - I've written about them
  - I've been focused on database structures, internals, and performance for hundreds of clients using SQL Server for over 20 years
- **I've worked on the SQL Server Development Team**
- **Since 1995, I've been independent of Microsoft working with customers to solve their problems**
- **I teach about SQL Server at all levels around the world**
- **My areas of expertise: architecture, design, internals and performance**

Course Focus and Structure (1)

- **This course expects basic knowledge of database terminology**
  - E.g. database, transaction log, backup
- **Applies to all Editions of SQL Server 2005 onward, except where noted**
- **This course talks about the physical impact of database design in Microsoft SQL Server**
  - This course is not about modeling, normalization, or database design theory
- **All content and demos are shown on SQL Server 2012 but many of these concepts apply to other platforms; the implementation and physical characteristics are likely to vary as well as what's optimal on each**

Course Focus and Structure (2)

- **Module 2: Data types and row size**
- **Module 3: Data types and index size**
- **Module 4: Data types and query performance**
- **This is the beginning of how to implement scalable databases on Microsoft SQL Server**
- **This course is really a starting point for many of my Pluralsight courses**
  - The next two courses in this series, coming in 2013 are:
    - SQL Server: Optimizing Stored Procedure Performance
    - SQL Server: Optimizing Ad Hoc Statement Performance
  - And there will be more as there are many ways to improve your database design and performance!