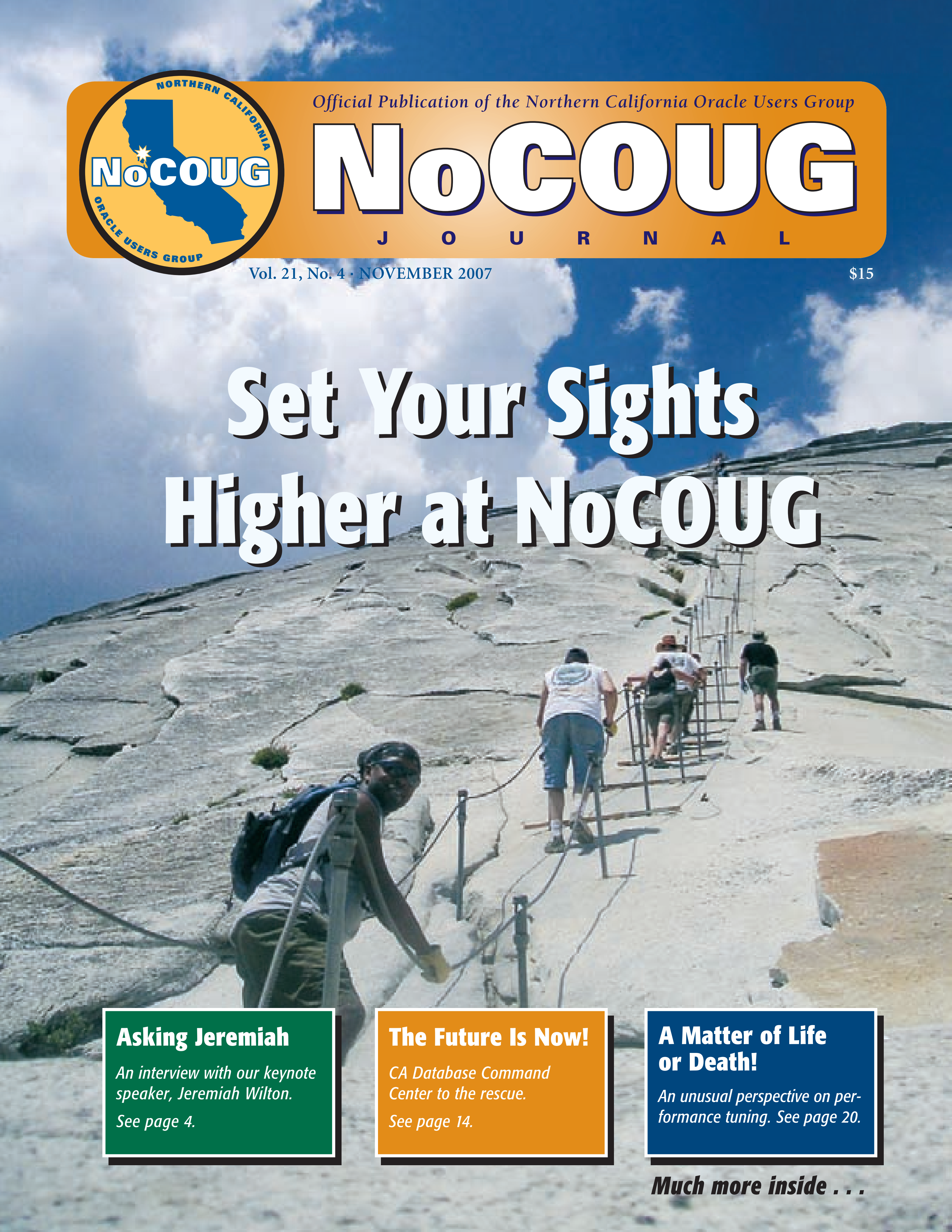
Official Publication of the Northern California Oracle Users Group

NoCOUG JOURNAL

Vol. 21, No. 4 · NOVEMBER 2007

$15



Set Your Sights Higher at NoCOUG

Asking Jeremiah

An interview with our keynote speaker, Jeremiah Wilton.

See page 4.

The Future Is Now!

CA Database Command Center to the rescue.

See page 14.

A Matter of Life or Death!

An unusual perspective on performance tuning. See page 20.

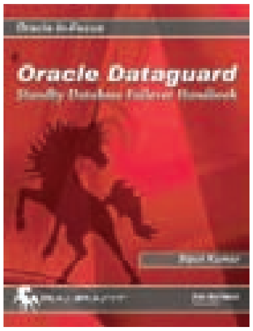
Much more inside . . .

Oracle Dataguard: Standby Database Failover Handbook

Reviewed by Brian Hitchcock

Details

Author: Bipul Kumar
ISBN: 0974599387
Format: Paperback
Pages: 230
Year of Publication: 2005
Edition: 1
Price: $27.95
Publisher: Rampant Techpress



Overall Review

The best reason to read an Oracle book is that you actually need the information. This also provides a good motivation to review such a book. Since I was (suddenly) tasked with supporting Data Guard for several of our Oracle environments, I was motivated to read carefully and put into action what I learned.

From this viewpoint I believe this book offers a good value. It isn't as thorough as you might like, but it gets to the point quickly and doesn't take a long time to read. When I was looking for a book about Data Guard, I didn't find very many. Other than this title, there is a section in the Oracle Press book *Oracle Database 10g High Availability with RAC, Flashback, and Data Guard*. I chose to focus on Data Guard only. While this book is somewhat out of date (published in 2005), that may have helped me, because the book covers how Data Guard 9*i* works and then adds the new features of 10*g*. Since most DBAs I know (but would they admit to knowing me?) support many Oracle databases that aren't 10*g* (yet?), I think covering both 9*i* and 10*g* would be valuable to most of the intended audience.

Note that using Data Guard means you are tying two (or more) databases together. This means you will need to review your support and monitoring procedures to reflect the new dependencies between primary and standby databases, as well as how to handle failover and switchover. This book does not cover these issues.

Chapter 1—Introduction to Oracle Data Guard

This introduction offers a review of failover options from 1990 to the present, including the options besides Data Guard such as Oracle Parallel Server, Replication, RAC, and Streams. The discussion includes the issue of Oracle licensing, which is an example of the kind of practical information available in this book. The pros and cons of manual standby using scripts (which avoids licensing costs) are discussed. Standby database concepts are presented, covering when the standby is read-only and a history of the specific Oracle standby database products in each of the major Oracle database versions 7.3 through 9*i*. A later chapter updates this for Oracle 10*g*. A detailed description of physical standby versus logical standby, and a comparison of the advantages and disadvantages of Data Guard including cost (hardware and software), are given next. This is followed by the operational constraints that Data Guard imposes, since you now have two databases that are intimately linked. The creation process is reviewed at a high level, including a sample configuration.

Chapter 2—Data Guard Architecture

This chapter provides a more detailed discussion of exactly how the process works as changes on the primary database are moved to the standby database. Each of the components of Data Guard is discussed, along with the critical role that the redo log plays in this process. The two ways that redo logs can be applied to the standby database are covered (redo apply and SQL apply) for the physical and logical standby respectively. A brief description is given of how Log Miner is used to extract SQL from the redo logs from the primary database. The issue of gaps in the sequence of redo logs from the primary is covered, as well as how Data Guard handles this. The architecture and components of Data Guard Broker are discussed. This is an Oracle utility that can be used to manage the Data Guard configuration from either the command line or as part of Enterprise Manager. *Failover*, the process in which the standby takes over when the primary fails, versus *switchover*, where the role of primary database can be switched back and forth between the primary and standby databases, is discussed. Switchover is a feature of Data Guard that you don't have if you are implementing a standby using your own scripts; it can be useful for testing failover and in certain upgrade scenarios.

Chapter 3—Implementing Standby Databases

Descriptions of a physical standby, logical standby, and setting up a standby using manual recovery—which means writing your own scripts to transport and apply archived redo logs to the standby—are covered. The physical standby is the more traditional standby in that it is constantly applying, or waiting to apply, the next archived redo log from the primary, and does not allow database users to connect even for read-only. You can, however, stop the process of applying archived redo logs and the physical standby will allow read-only database access. The physical standby is an exact copy of the primary. The logical standby is a newer feature whereby archived redo logs from the primary are shipped to the standby and LogMiner is used to extract the SQL statements that caused changes in the primary database. These SQL statements are then applied to the standby database. This means the process that applies the SQL statements is just like any other user connection to the database, and the standby database can be open for users at the same time it is applying changes from the primary. Data Guard will prevent users from making changes in the logical standby database of objects that are being maintained (guarded) by Data Guard.

The additional details of creating a physical standby on the same machine as the primary are covered. Basically, this means the standby must use different filesystems than the primary, and this requires setting up init.ora parameters that will map between the primary and standby filesystems.

The logical standby does have some limitations compared to the physical, and these are covered here as well. The logical standby does not support all the data types that the physical standby does.

Chapter 4—Standby Database Administration

Issues covered here include general administration and monitoring. For example, once you install Data Guard for a standby database, the primary and standby databases are intimately related. If the standby database becomes unreachable for any reason, an "oracle unavailable" message appears in the primary database alert log. Since you may have existing monitoring that scans the primary database alert log looking for errors and warnings, it can be very upsetting the first time you are paged because the primary database is "unavailable," only to find out that it is really just an informational message about the standby database, not the primary. Depending on the criticality of the standby, this specific message may or may not be critical.

General administration of Data Guard covers starting and stopping the process of applying archived redo logs to the physical standby database, starting and stopping the SQL apply process for a logical standby, and shutting down the standby altogether. Again, since the primary and standby are linked under Data Guard, if you simply shut down one or the other, you may get messages from the other database. You need to develop your own support processes to handle the new complexities of Data Guard. Also covered is the fact that triggers that fire in the primary database don't fire in the standby, and primary database constraints aren't enforced in the standby. How to skip tables is explained, which means that you can tell Data Guard not to replicate specific tables into the standby. This feature, combined with the constraints not being enforced, means you can configure a standby such that it doesn't maintain the referential integrity of the primary.

In the event of a switchover or failover, details of what must be done are presented. This depends on the particular scenario you are in, and there are several. You have to worry about archived redo logs that haven't been shipped to the standby or that haven't been applied to the standby, and so on.

This chapter also covers how you can set up cascading standby databases in which a primary database sends archived redo logs to a standby database, which then sends them to one or more additional standby databases.

Chapter 5—Log Management Services

This chapter covers details of how to configure various options for getting the archived redo logs to the standby, as well as protection modes. Protection modes refer to what Data Guard will do if the primary can't send an archived redo log to the standby. If you configure Maximum Protection and the standby isn't available, the primary database will eventually stop, since it can't reuse a redo log until it has been copied to the standby. This reminds me of two-phase commit, but I doubt anyone else is old enough to remember that! Maximum Protection mode is only available for the physical standby. Maximum Performance, which is the default, means the primary database will try to send the archived redo log to the standby, and if it can't, it keeps going. A third available option is covered: Maximum Availability, which is a combination of the two previous options. In this configuration, the primary will try to write to both the local and standby archived redo log destination, but if it can't, it will move on. However, unlike Maximum Performance, as soon as the primary can communicate with the standby again, it will, and it will ensure that the archived redo logs are sent to both primary and standby as long as possible.

This chapter also covers the options for the shipping of the archived redo logs themselves. By default, Data Guard will wait until the archived redo log is complete on the primary before shipping it to the standby. You can also configure what is called *standby* redo logs on the standby database, and Data Guard will ship the entries from the primary archived redo log just as fast as they are written to the primary.

> ***"I have seen reviews of books that say the book is simply a rehash of the manuals. By definition, to some extent, this will always be the case, assuming that the Oracle manuals are any good at all. However, to make this criticism is like saying you don't need a map of San Francisco because it's just a rehash of the globe. True, but if you need to find the NoCOUG conference at the PG&E building in the rain, you will find a map of San Francisco very useful."***

An interesting detail of the logical standby is that since it is a normal database that is open for user access as well as applying the SQL from the primary database, it is also generating its own archived redo logs. This can be confusing. On the standby database server, you need to manage disk space for both the archived redo logs shipped from the primary database and the archived redo logs that are generated by the logical standby database itself. Both of these locations on disk can run out of disk space, and you need to be backing up the archived redo logs of the logical standby database to support recovery. This is critical if you add any database object to the logical standby database. This is a feature of the logical standby. You can add schemas—for reporting, for example—that are not present in the primary. These additional schemas might be used to store aggregates of primary table data, and so on.

Chapter 6—Switchover and Failover

An advantage of Data Guard over your own scripts that maintain a standby database is the ability to switchover. This means the primary and standby databases can swap roles back and forth. This is useful for testing failover and upgrades. When using scripts to maintain the standby, you can only failover, which means that the standby database is opened. After this, you can't switch back; i.e., the former standby database, which is now the primary, can't become a standby again without recovering the previous primary database and then rebuilding the former standby from the recovered standby. For Data Guard, failover means the same thing, but by using switchover, the primary and standby databases can swap roles repeatedly.

This chapter covers the details of the role transition process, by which switchover (or failover) is accomplished. This includes details for transitioning to a logical standby versus the more typical physical standby. Note that switch-over (or failover) to a logical standby, such that the logical standby becomes the primary database, can be problematic. Recall that a logical standby has limitations, such as non-support of some data types. If the primary database uses any of the data types that Data Guard doesn't support, you can't switchover (or failover) to the logical standby because it can't function as a complete re-creation of the primary database.

Chapter 7—Performance Tuning of Data Guard Configuration

When using Data Guard you need to be aware of the possible performance impacts on the primary database, as well as the possible impact on the standby database users if the standby isn't kept up-to-date relative to the business requirements. Monitoring Data Guard performance focuses on the log transfer service, which gets the archived redo logs from the primary to the standby, and the log apply services, which apply the transferred redo logs to the standby database.

Scripts are provided to get performance metrics from various views for monitoring the progress of both redo log transfer and the redo log apply processes.

If the Data Guard environment is configured for maximum protection, the primary database may be forced to stop if there is a significant delay in transferring and applying the redo logs to the standby.

From the perspective of the standby database, performance means getting out of synch with the primary. This requirement will be defined by the business requirements. A standby that must be up-to-date within 5 minutes of the primary at all times will have different performance issues than a standby that is not updated during business hours and must catch up each night. Clearly the network bandwidth available is a factor, as is the use of logical standby, which must scan the redo logs and apply individual SQL statements.

Specific tuning advice for the log transfer process covers network bandwidth, TNS parameters, and I/O issues on the standby server. For the log apply services, the tuning advice focuses on the logical standby and covers LogMiner memory usage, impact of full table scans (caused by the primary table lacking a primary or unique key), transaction consistency, and shared pool size.

Chapter 8—Data Guard Broker

This chapter covers the server-side and client-side components provided by Oracle to help monitor (and create) Data Guard environments. There are command-line and Enterprise Manager versions of each component. The process of using these components to configure, monitor, and maintain a Data Guard environment is covered. Differences between the 9*i* and 10*g* features of the Broker are presented. Data Guard Broker can also be used to execute the role transition between primary and standby databases.

Chapter 9—Recovery Manager and Data Guard

Since building the Data Guard standby database starts with obtaining a backup of the primary database, RMAN can be instrumental in administering a Data Guard environment. Incremental RMAN backups can be very useful in reducing the time needed to create a standby for a very large primary database. RMAN terminology and the process of creating a physical standby database using RMAN are covered.

RMAN can also be used to back up the standby so that backups don't have to be made on the primary. Details of how to recover the primary database from RMAN backups made on the standby are covered. While this chapter is not a complete reference for RMAN, it points out the advantages of using RMAN and identifies the main issues that must be addressed.

Chapter 10—Oracle 10*g*: New Features of Data Guard

This chapter covers specific changes made between Data Guard 9*i* and 10*g*. Examples of what is covered include real-time apply and simplified creation of a logical standby. Real-time apply means the log apply services don't need to wait for a complete archived redo log from the primary before applying the redo log information to the standby. This means the standby can be kept more up-to-date relative to the primary. This feature uses standby redo logs, which are created on the standby database. Data Guard 10*g* makes creation of a logical standby easier by not requiring a shutdown of the primary database. Syntax changes allow creation of the logical standby control file, which contains the information needed to establish the logical database without a primary database shutdown.

Appendix A—Data Guard Reference

This covers init parameters, dynamic performance views, and PL/SQL packages used by Data Guard.

Appendix B—Troubleshooting Guide

This presents procedures to solve common issues for log transfer services, SQL apply operations, and missing temp files, among others.

Conclusion

As with the other book I've read from Rampant (*Oracle Performance Troubleshooting with Dictionary Internals SQL & Tuning Scripts, Oracle In-Focus Series* by Robin Schumacher, published by Rampant TechPress in 2003; see **www.brianhitchcock.net** for a review), this book is a good value. You learn what the author has to tell you in an efficient manner. Yes, some detail is left out, but I liked the fact that I could get through the book quickly. I have seen reviews of books that say the book is simply a rehash of the manuals. By definition, to some extent, this will always be the case, assuming that the Oracle manuals are any good at all. However, to make this criticism is like saying you don't need a map of San Francisco because it's just a rehash of the globe. True, but if you need to find the NoCOUG conference at the PG&E building in the rain, you will find a map of San Francisco very useful. For my purposes, I needed a quick overview of how Data Guard works, what the basic options are, and how to set it up. I got all of that from this book.

The writing could use some further editing. There are more apparent typos than I am used to in the Oracle Press books, but this didn't detract from the value of the information. I would have benefited from more diagrams that showed in detail how the redo information flows between primary and standby, and I would have benefited from more details about the setup and refresh processes.

A final testament to the value of what I learned from this book is that I was able to review the architecture that had been put in place by the consultants and offer an alternative approach that was simpler for my group to support. The consultants agreed that the alternative approach would work, so I believe I got what I needed from this book. ▲

About the Author(s)

Bipul Kumar is a senior application engineer for a telecommunications company. He is responsible for Oracle database administration, performance tuning, recovery, and application development. He has more than six years of experience in creating and managing Oracle databases of varying sizes, primarily on Unix operating systems.

Brian Hitchcock has worked at Sun Microsystems in Newark, California, for the past 11 years. He is a member of a DBA team that supports 2400+ databases for many different applications at Sun. He frequently handles issues involving tuning, character sets, and Oracle applications. Other interests include Formula One racing, finishing his second Tiffany Wisteria lamp, Springbok puzzles, Märklin model trains, Corel Painter 8, and watching TV (TiVo rules!). Previous book reviews by Brian and his contact information are available at **www.brianhitchcock.net**.