Chapter No.3
"Design Considerations"
In this package, you will find:
A Biography of the author of the book
A preview chapter from the book, Chapter NO.3 "Design Considerations"
A synopsis of the book’s content
Information on where to buy this book

About the Author

John P. Jeffries has lived in the southeast of England for most of his life. Although predominantly based in London, he has worked in many countries around the world. He enjoys travel and is the Director of his own consultancy business: www.spirotek.co.uk.

Originally from a development background, he has worked for a number of global software companies including Oracle Corporation and Siebel Systems. His time at Siebel was spent designing and developing ETL solutions for Data Warehouses, ultimately enhancing the eBilling and Billing Analytics products with an Oracle Warehouse Builder based API. He spent six years working in Oracle Consulting in EMEA and Oracle Advanced Customer Services in the UK as a Senior Principal Consultant, earning the internal title of "The UK's Data Replication Expert". Now a freelance consultant, the author can be found onsite in many of the world's most respected financial institutions in London, consulting on Oracle GoldenGate, Streams, and Active Data Guard. With over 15 years of Oracle experience, and an OCP since Oracle 8i, the author has extensive knowledge of Oracle databases, including Enterprise Linux and RAC, coupled with the ability to design and build high performance distributed database systems. He has trained internal and external clients in Data Warehousing and Data Replication techniques, and

For More Information:
continues to share his knowledge and experience through his own website:

Thank you for purchasing my book, which would not have been possible without the help and support from a number of key individuals and organizations.

Firstly I wish to thank my wife, Wendy, for both her drive and encouragement. Also the sacrifice that she has endured, putting aside other commitments to support me in my accomplishment.

Secondly, I wish to thank my friend and colleague, Paul Vale for his depth of knowledge of Oracle Streams. I worked closely with Paul for two years on an extensive data replication project where our shared knowledge and experience inspired me to write a book on Oracle GoldenGate.

Finally, I wish to thank Packt Publishing for agreeing to work with me and publish this book. Even now, I am astounded by the lack of available technical material on GoldenGate; this being another reason for writing. I trust you will find this book both interesting and informative, helping you to successfully implement an Oracle 11g GoldenGate environment.

For More Information:
Oracle GoldenGate 11g Implementer's guide

Data replication is an important part of any database system that is growing due to today's demand for real-time reporting and regulatory requirements. GoldenGate has recently become Oracle's strategic real-time data replication solution. Until now, very little has been written about how to implement GoldenGate in a production enterprise environment where performance, scalability, and data integrity are paramount.

Your days of dismay over the lack of documentation over Oracle GoldenGate are over. Welcome to Oracle GoldenGate 11g Implementer's guide—a comprehensive, practical book that will deliver answers to your questions in a clear, concise style, allowing you to progress effectively in a timeline-driven environment. Based on the author's own experience, this long awaited GoldenGate administration book has all that is required to install, design, configure, and tune data replication solutions suited to every environment. Be the first to master GoldenGate's power and flexibility by reading this unique hands-on implementation companion.

Computers need to send data from one system to another in a timely manner to satisfy the ever-increasing need for speed. Regardless of whether you are a novice or an expert—or someone in between—this book will guide you through all the steps necessary to build a high-performance GoldenGate solution on Oracle11gR1. Expert users can dive into key topic areas such as performance tuning or troubleshooting, while novice users can step through the early installation and configuration chapters, later progressing to the advanced chapters.

This book is more than an implementation guide. It offers detailed real-life examples, encouraging additional thought and discussion by going beyond the manual.

With Oracle GoldenGate 11g Implementer's guide in hand, you'll be designing, installing, and configuring high-performance solutions using GoldenGate within minutes.

What This Book Covers

Chapter 1, Getting Started, provides an introduction to Oracle GoldenGate, inspiring thought by describing the key components, processes, and considerations required to build and implement a GoldenGate solution.

The topics covered the evolution of GoldenGate Software, including the architecture behind the technology followed by the solutions GoldenGate can provide, coupled with effective design.

For More Information:
Chapter 2, *Installing and Preparing GoldenGate*, walks you through the individual tasks needed to complete an end-to-end GoldenGate installation, including the download of the software through to creating a simple data replication environment.

Chapter 3, *Design Considerations*, addresses some of the issues that influence the decision making process when designing a GoldenGate solution. These include design considerations for both performance and high availability. Here you can choose the appropriate hardware and topology to deliver a fast, robust and scalable solution.

Chapter 4, *Configuring Oracle GoldenGate*, initially discusses the main GoldenGate parameters, and provides a methodical approach to the configuration process, stepping through each task to give the depth of information necessary to successfully implement GoldenGate on Oracle 11g. By providing the building blocks, this chapter forms the basis for more complex configurations.

Chapter 5, *Configuration Options*, looks at the available options that allow your configuration to extend in functionality and performance. We start with performance enhancements, later exploring the security features, including data compression and encryption, and finally discussing the options available to implement DDL replication.

Chapter 6, *Configuring GoldenGate for HA* teaches you how to configure Oracle GoldenGate in a RAC environment and explores the various components that effectively enable HA for data replication and integration.

Chapter 7, *Advanced Configuration*, gives you a deeper understanding of GoldenGate configuration. By the time you are done with this chapter, you will be able to explore and realize each parameter specification and further develop your GoldenGate configuration.

Chapter 8, *Managing Oracle GoldenGate*, focuses on the management features already built in to the GoldenGate Command Interpreter (GGSCI). We discuss and implement a number of utilities, including tips and tricks that allow you to manage your GoldenGate environment effectively at no extra cost. This includes a Perl script that will format the text output from the GGSCI "INFO ALL" command so that the data can be loaded into MS Excel for graphing and trend analysis.

Chapter 9, *Performance Tuning*, focuses on the main areas that lend themselves to tuning, especially parallel processing and load balancing, enabling high data throughput and very low latency. Performance enhancing new features that are available in the next version of GoldenGate are also discussed and evaluated.

For More Information:  
Chapter 10, Troubleshooting GoldenGate, provides a troubleshooting guide for Oracle GoldenGate. We address some of the common issues faced by the implementer, followed by a discussion of the tools and utilities available to help resolve them. This includes the creation of an Exception Handler that automatically logs the details of a failed transaction for further analysis and resolution.

This book has three appendices; their purpose is to provide a "quick reference" as well as address any subjects and terminology not addressed in the previous chapters.

Appendix A, GGSCI Commands, provides a quick reference guide to all the available GoldenGate Software Command Interface (GGSCI) commands.

Appendix B, GoldenGate Installed Components, lists the GoldenGate installed components and their description.

Appendix C, The Future of Oracle GoldenGate, discusses Oracle's strategic approach to data replication and integration. The latest release of GoldenGate including its sister products are described along with links to official documentation and forums on the Oracle website.

For More Information:
Design Considerations

The most important step in any IT development project is the design. This must be the first step as changes to the design at a later stage will cost time and money. Get the design right and your system will continue to perform well as the user base increases.

At a high level, the design must include the following generic requirements:

- Hardware
- Software
- Network
- Storage
- Performance

All the above must be factored into the overall system architecture. So let’s take a look at some of the options and the key design issues.

In this chapter, we will discuss the following areas to help answer some of the questions that influence the decision making process.

- Methods for replicating data: What do we want to achieve?
- Networking: How do we make our solution resilient to network outages?
- Performance and scalability: Will our solution support future non functional requirements?
- Backup and recovery: Can we restore service quickly with no data loss?
- Hardware: What is the size of our budget? Do we need powerful servers? Can we cluster lower spec machines?

For More Information:
Replication methods

So you have a fast reliable network between your source and target sites. You also have a schema design that is scalable and logically split. You now need to choose the replication architecture; One to One, One to Many, active-active, active-passive, and so on. This consideration may already be answered for you by the sheer fact of what the system has to achieve. Let’s take a look at some configuration options.

Active-active

Let’s assume a multi-national computer hardware company has an office in London and New York. Data entry clerks are employed at both sites inputting orders into an Order Management System. There is also a procurement department that updates the system inventory with volumes of stock and new products related to a US or European market. European countries are managed by London, and the US States are managed by New York. A requirement exists where the underlying database systems must be kept in synchronisation. Should one of the systems fail, London users can connect to New York and vice-versa allowing business to continue and orders to be taken. Oracle GoldenGate’s active-active architecture provides the best solution to this requirement, ensuring that the database systems on both sides of the pond are kept synchronised in case of failure.

Another feature the active-active configuration has to offer is the ability to load balance operations. Rather than have effectively a DR site in both locations, the European users could be allowed access to New York and London systems and vice-versa. Should a site fail, then the DR solution could be quickly implemented.
Active-passive

The active-passive bi-directional configuration replicates data from an active primary database to a full replica database. Sticking with the earlier example, the business would need to decide which site is the primary where all users connect. For example, in the event of a failure in London, the application could be configured to failover to New York.

Depending on the failure scenario, another option is to start up the passive configuration, effectively turning the active-passive configuration into active-active.
Cascading

The Cascading GoldenGate topology offers a number of "drop-off" points that are intermediate targets being populated from a single source. The question here is "what data do I drop at which site?" Once this question has been answered by the business, it is then a case of configuring filters in Replicat parameter files allowing just the selected data to be replicated. All of the data is passed on to the next target where it is filtered and applied again.

This type of configuration lends itself to a head office system updating its satellite office systems in a round robin fashion. In this case, only the relevant data is replicated at each target site. Another design, already discussed in Chapter 1, is the Hub and Spoke solution, where all target sites are updated simultaneously. This is a typical head office topology, but additional configuration and resources would be required at the source site to ship the data in a timely manner. The CPU, network, and file storage requirements must be sufficient to accommodate and send the data to multiple targets.

For More Information:
Physical Standby

A Physical Standby database is a robust Oracle DR solution managed by the Oracle Data Guard product. The Physical Standby database is essentially a mirror copy of its Primary, which lends itself perfectly for failover scenarios. However, it is not easy to replicate data from the Physical Standby database, because it does not generate any of its own redo. That said, it is possible to configure GoldenGate to read the archived standby logs in Archive Log Only (ALO) mode. Despite being potentially slower, it may be prudent to feed a downstream system on the DR site using this mechanism, rather than having two data streams configured from the Primary database. This reduces network bandwidth utilization, as shown in the following diagram:

Reducing network traffic is particularly important when there is considerable distance between the primary and the DR site.

For More Information:
Design Considerations

Networking
The network should not be taken for granted. It is a fundamental component in data replication and must be considered in the design process. Not only must it be fast, it must be reliable. In the following paragraphs, we look at ways to make our network resilient to faults and subsequent outages, in an effort to maintain zero downtime.

Surviving network outages
Probably one of your biggest fears in a replication environment is network failure. Should the network fail, the source trail will fill as the transactions continue on the source database, ultimately filling the filesystem to 100% utilization, causing the Extract process to abend. Depending on the length of the outage, data in the database's redologs may be overwritten causing you the additional task of configuring GoldenGate to extract data from the database's archived logs. This is not ideal as you already have the backlog of data in the trail files to ship to the target site once the network is restored. Therefore, ensure there is sufficient disk space available to accommodate data for the longest network outage during the busiest period.

Disks are relatively cheap nowadays. Providing ample space for your trail files will help to reduce the recovery time from the network outage.

Redundant networks
One of the key components in your GoldenGate implementation is the network. Without the ability to transfer data from the source to the target, it is rendered useless. So, you not only need a fast network but one that will always be available. This is where redundant networks come into play, offering speed and reliability.

NIC teaming
One method of achieving redundancy is Network Interface Card (NIC) teaming or bonding. Here two or more Ethernet ports can be "coupled" to form a bonded network supporting one IP address. The main goal of NIC teaming is to use two or more Ethernet ports connected to two or more different access network switches thus avoiding a single point of failure. The following diagram illustrates the redundant features of NIC teaming:

For More Information:
Linux (OEL/RHEL 4 and above) supports NIC teaming with no additional software requirements. It is purely a matter of network configuration stored in text files in the 
/etc/sysconfig/network-scripts directory. The following steps show how to configure a server for NIC teaming:

1. First, you need to log on as root user and create a bond0 config file using the vi text editor.
   ```
   # vi /etc/sysconfig/network-scripts/ifcfg-bond0
   ```

2. Append the following lines to it, replacing the IP address with your actual IP address, then save file and exit to shell prompt:
   ```
   DEVICE=bond0
   IPADDR=192.168.1.20
   NETWORK=192.168.1.0
   NETMASK=255.255.255.0
   USERCTL=no
   BOOTPROTO=none
   ONBOOT=yes
   ```

3. Choose the Ethernet ports you wish to bond, and then open both configurations in turn using the vi text editor, replacing ethn with the respective port number.
   ```
   # vi /etc/sysconfig/network-scripts/ifcfg-eth2
   # vi /etc/sysconfig/network-scripts/ifcfg-eth4
   ```

4. Modify the configuration as follows:
   ```
   DEVICE=ethn
   USERCTL=no
   ONBOOT=yes
   MASTER=bond0
   SLAVE=yes
   BOOTPROTO=none
   ```
5. Save the files and exit to shell prompt.
6. To make sure the bonding module is loaded when the bonding interface (bond0) is brought up, you need to modify the kernel modules configuration file:
   
   ```
   # vi /etc/modprobe.conf
   ```

7. Append the following two lines to the file:
   ```
   alias bond0 bonding
   options bond0 mode=balance-alb miimon=100
   ```

8. Finally, load the bonding module and restart the network services:
   ```
   # modprobe bonding
   # service network restart
   ```

You now have a bonded network that will load balance when both physical networks are available, providing additional bandwidth and enhanced performance. Should one network fail, the available bandwidth will be halved, but the network will still be available.

Non-functional requirements (NFRs)

Irrespective of the functional requirements, the design must also include the non-functional requirements (NFR) in order to achieve the overall goal of delivering a robust, high performance, and stable system.

Latency

One of the main NFRs is performance. How long does it take to replicate a transaction from the source database to the target? This is known as end-to-end latency that typically has a threshold that must not be breeched in order to satisfy the specified NFR.

GoldenGate refers to latency as lag, which can be measured at different intervals in the replication process. These are:

- **Source to Extract**: The time taken for a record to be processed by the Extract compared to the commit timestamp on the database
- **Replicat to Target**: The time taken for the last record to be processed by the Replicat compared to the record creation time in the trail file

A well designed system may encounter spikes in latency but it should never be continuous or growing. Trying to tune GoldenGate when the design is poor is a difficult situation to be in. For the system to perform well you may need to revisit the design.

---

For More Information:

Availability

Another important NFR is availability. Normally quoted as a percentage, the system must be available for the specified length of time. An example NFR of 99.9% availability equates to a downtime of 8.76 hours a year, which sounds quite a lot, especially if it were to occur all at once.

Oracle's maximum availability architecture (MAA) offers enhanced availability through products such as RAC and Dataguard. However, as we have previously discussed, the network plays a major role in data replication. The NFR probably relates to the whole system, so you need to be sure your design covers all components.

We look at configuring GoldenGate on Real Application Clusters (RAC) as a MAA solution in the Chapter 6, Configuring GoldenGate for High Availability.

Backup and recovery

Equally important as the other NFRs is the recovery time. If you cannot recover from a disaster your system will be unavailable or worse still, data will be lost. GoldenGate prides itself on robustness, having proven to the industry that zero downtime data migrations are possible whilst users are still online!

Of course, you need to backup your source and target databases regularly using a product such as Oracle Recovery Manager (RMAN) — typically performing a full backup once a week on a Sunday with incremental backups running each night over a 7 day window, but this is not the full story. We need to consider backing up the GoldenGate home and sub-directories that contain the trail files, checkpoint files, and so on. Without these, GoldenGate could not recover from the last checkpoint and a new initial load would be required. RMAN (11gR1) will not back up OS or non-database files so either use UNIX shell commands or a third party product such as Veritas NetBackup.

You may decide to place your GoldenGate sub-directories on shared storage such as a Storage Area Network (SAN) where data resilience is automatically maintained. This may be the best design solution given that the disks are shared and the available speed of recovery. For example, restoring data from EMC SnapView.

The best recovery solution is the Disaster Recovery (DR) site, where you can quickly switchover or failover to a backup system. GoldenGate may already be part of this strategy, so ensure your DR solution is robust by scheduling regular backups as previously discussed.
Design Considerations

The following example architecture diagram helps to illustrate the design:

![Example Architecture Diagram]

Although a key area in the overall design B & R is sometimes overlooked. Oracle provides a number of HA solutions that offer fast and reliable mechanisms to ensure your data is not only backed up but always available. In this chapter, we have already discussed Oracle RAC, Dataguard and RMAN, with GoldenGate we have a fourth member of the MAA team.

Hardware considerations

Hardware is one of the most important components in the overall design. Without the appropriate memory footprint, CPU power, network, or storage solution our design will fail before the project even "gets off the ground". Another consideration is how to arrange the hardware to obtain the best performance and scalability. Let's take a look at a few options.

Computing architectures

The configuration and arrangement of hardware components is known as the system architecture. This section concentrates on the common computer architectures available and discusses their advantages and disadvantages for distributed database network supporting a Web-based application.

The following diagram shows the typical hardware components required:

For More Information:
GoldenGate is deemed as middleware and normally runs on the database tier where the software can access the database and it's online redo logs.

**Grid computing**
Nowadays, highly powerful servers can be procured preconfigured to the customer's specification. One of the most cost effective is the x86 64 bit Linux server, which offers high performance at a budget price. Should your system require more "horse power", you just add more servers. This is known as Grid computing.

Grid computing offers multiple applications to share computing infrastructure, resulting in greater flexibility, low cost, power efficiency, performance, scalability and availability, all at the same time.

What does this mean for GoldenGate? As we know, GoldenGate runs a number of local processes. Should we wish to synchronize data between more than one server we would have to adopt a shared storage solution or maybe in-memory caching such as Oracle Coherence. This all adds to the overall complexity of the design.

**Single server**
A single database and application server may also be a low cost option. However, it is not scalable and will cost you a lot more having to upgrade the hardware as the number of users increase. On the other hand, the single server option does provide simplicity and can be easily configured and maintained. The choice in architecture is largely dependent on the application, not just the cost!
Clusters

There is nothing new about clustered environments; they have existed for years. DEC VMS is a classic example. Clustered environments have proven their worth over the years; for example, Oracle Real Application Clusters (RAC) has been around since Oracle 9i and before that in Oracle 8 it was known as Oracle Parallel Server (OPS). Today, RAC is one of the most common database architectures offering performance, resilience, and scalability at a reasonable cost (depending on the number of nodes and CPU cores). RAC does however demand shared storage, which may add to the cost.

As with Grid computing, GoldenGate requires shared storage in order to replicate data from more than one thread or instance.

Machines

Although GoldenGate is multi-platform and supports heterogeneous databases, this book focuses on Linux and Oracle. In a production environment, the database server is typically a powerful machine costing thousands of dollars. However, the hardware investment can be significantly reduced without compromising performance by using the Linux x86-64 operating system on Intel based machines with 64 bit architecture.

The x86-64 Linux Server

The x86-64 Linux Server is essentially a PC having 64 bit Red Hat or Oracle Enterprise Linux installed. Typically, a standard Linux business server delivered from HP or Dell would have the following minimum hardware specification:

- 4 Quad-Core Intel or AMD processors
- 16 GB Memory
- 2 x 500 GB Hard disks (possibly a mirrored pair)
- 4 x Gigabit Ethernet Ports

Depending on the application, the hardware specification is more than adequate for clustered or grid environments, but may not be sufficient as a single server. We must also consider the different layers involved in a computer system, from the application and middleware layers up to the database tier, which includes the storage.

How many servers are you going to allocate to each layer?

Good question. It is common to have more application servers than database servers supporting the system. The application servers providing the "user experience" offering the functionality and response times through load balancing and caching technologies, ultimately querying, or modifying data in the database before rendering and serving Web pages.

For More Information:

The Database Machine
Since Oracle's acquisition of Sun Microsystems in January 2010, the corporation has marketed the "Database Machine" and Exadata2. At the time of writing, the minimum specification Database Machine (half rack) comprises of two database servers having the following:

- 2 Quad-Core Intel Xeon E5540 Processors
- 72 GB Memory
- Disk Controller HBA with 512MB Battery Backed Write Cache
- 4 x 146 GB SAS 10,000 RPM disks
- Dual-Port QDR InfiniBand Host Channel Adapter
- 4 Embedded Gigabit EthernetPorts

Plus the shared storage:

- 3 x Sun Oracle Exadata Storage Servers with 12 x 600 GB 15,000 RPM SAS disks or
- 12 x 2 TB 7,200 RPM SATA disks
- Including 1.1 TB Exadata Smart Flash Cache

The Database Machine is purely supporting the database tier, uniquely designed for Oracle databases, offering very high speed transactional processing capabilities with Exadata V2 storage. This is undoubtedly the "Ferrari" of all database servers, which has proved to be a highly successful product for Oracle. However, Ferrari's are not cheap and nor are Oracle Sun Database Machines! Your design therefore needs to balance the costs against acceptable performance and scalability.

Scaling up and out
Probably one of the most difficult decisions to make in your overall design is whether to scale up or scale out. If you choose to scale up and use a single powerful database server with expansion for additional CPU and memory, this may prove to be a short term solution. Eventually, your application's performance will "hit the buffers" where the database server has no more capacity to scale. To resolve this problem by replacing the server with an even more powerful machine would incur significant costs.

So is the answer to scale out? Not necessarily, scaling out is not without its problems. Considering Oracle RAC as a clustered database solution, where multiple instances hosted on multiple nodes all connect to the same database on shared storage, we move into a world of global cache locking and waits.

For More Information:
Consider a user connected to instance 1 on node A executing a query that causes a full table scan against a long table having over 1 million rows. To try to reduce the cost of I/O, Oracle will look at the global cache to see if the blocks reside in memory. However, had a similar query been executed on a remote node, Oracle will transfer the blocks across the interconnect network between the nodes which may be a slower operation than scanning the underlying disk subsystem. For this reason and to reduce the potential performance overhead, it is possible to configure a 2 node Oracle RAC system in active-passive mode. Here, one instance takes the load while the other instance becomes the redundant standby. But we are back to one node again!

The key is to find the right balance. For example, you would not want to overwhelm your database servers with requests from an enormous array of application servers. The application response time would suffer as the database becomes the bottleneck. It is a case of tuning the system to achieve the best performance across the available hardware. By all means leverage the advantages of Oracle RAC on low cost servers, but make sure your application's SQL is optimized for the database. Look at schema design, table partitioning, even instance partitioning where data can be grouped across the nodes. An example of this would be countries. Users from the UK connect to instance 1, French users to instance 2, German users to instance 3, and so on.

What is important for GoldenGate is the database redo logs. These need to be on shared storage on fast disks and accessible to the Extract processes. It is an Oracle best practice to place the database's online redo logs on fast disks that are striped and mirrored (not RAID 5), because the database is constantly writing to them. RAID5 is slower on write operations as it is not recommended for redo logs. Furthermore, if the I/O is slow, the database performance will suffer as a result of the 'logfile sync' wait event.

**Changed data management**

It's all very well replicating data in real-time between a source and target database, but what if something disastrous happens to the source data; a user drops a table or deletes important records or the data becomes corrupt? The answer maybe to build a delay into the changed data delivery but this seems to defeat the object of fast data replication. What can be done to manage the changed data, ensuring that only the valid transactions succeed?

There are a number of solutions to this problem, none of which prevent user error or data corruption. Let's now take a look at some of these, which are provided by the Oracle database.
Point in Time Recovery (PITR)

Since Oracle 10g, the database provides "flashback" technology allowing the database, a table or even a transaction to be flashed back to a SCN or timestamp. The Flashback technology provides a fast reliable recovery mechanism over the traditional method of performing a point in time recovery.

Oracle Recovery Manager (RMAN)

RMAN supports PITR. However, the database would have to be mounted and not open preventing users from connecting. Furthermore, the database would need to be restored from a backup and then recovered to a specified SCN or timestamp. All this takes time and is unacceptable, particularly with the database offline!

Flashback

A far quicker recovery method is the Oracle 11g Flashback technology. Here, a dropped table can be recovered instantaneously from the Recycle Bin by one command and with the database open. It is a similar story for individual transactions too. These can be backed out using the information provided from a Flashback Transaction Query. The result of adopting these methods would also generate redo that the GoldenGate Extract process would then write to the trail files for onward replication. Therefore, no action is required on the target database.

Should you wish to flashback the whole database to a point in time before the error, Flashback would need to have been enabled at database level. This operation causes the database to generate flashback logs in addition to its redo logs, all of which are written to the Flash Recovery Area (FRA). To recover from data loss or corruption in a GoldenGate environment, it is important to perform the flashback on both source and target databases. This is however, an off-line operation. The GoldenGate Veridata product can be used to perform the data comparison following recovery.

To guard against human error, Flashback technology appears to provide the solution, but what does this mean for GoldenGate? Simply alter your Extract process to start replicating data from the specified timestamp.

For a bit of insurance and peace of mind, it's worth enabling Flashback on your mission critical source and target databases, making sure you factor in the additional storage requirements for the FRA in your design.

For More Information:
Design Considerations

SAN Snapshot
SAN Snapshots provide an alternative solution to PITR. Typically, snaps are scheduled every 30 minutes, capturing a "snapshot" of the data that will provide a restore point. The beauty of the SAN Snapshot is its ability to snap all file types as an online operation; database files, raw files, filesystems etc, which lends itself perfectly to a GoldenGate environment. You no longer need to concern yourself with winding back your database and your Extract process to a point in time, just restore the snap and wind forward to just before the error by applying the database archived logs. The GoldenGate trail and checkpoint files will remain in synchronization with the database as if nothing ever happened. The only issue you may face is having to manually replay any legitimate transactions that occurred after the error or corruption.

Summary
In this chapter, we revisited the GoldenGate topology, this time from a design viewpoint. We discussed the different replication methods and architectures and the hardware associated with each one. Functional and non-functional requirements were evaluated against the cost of hardware resources, encouraging a mindful balance in the overall design. Finally, we concentrated on Backup and Recovery options that are often overlooked, but have huge importance in your GoldenGate implementation.

In the next chapter, we delve a little deeper into the configuration detail, discussing the range and scope of available options and functions that make GoldenGate such a powerful and feature-rich product.

For More Information:
Where to buy this book


Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.