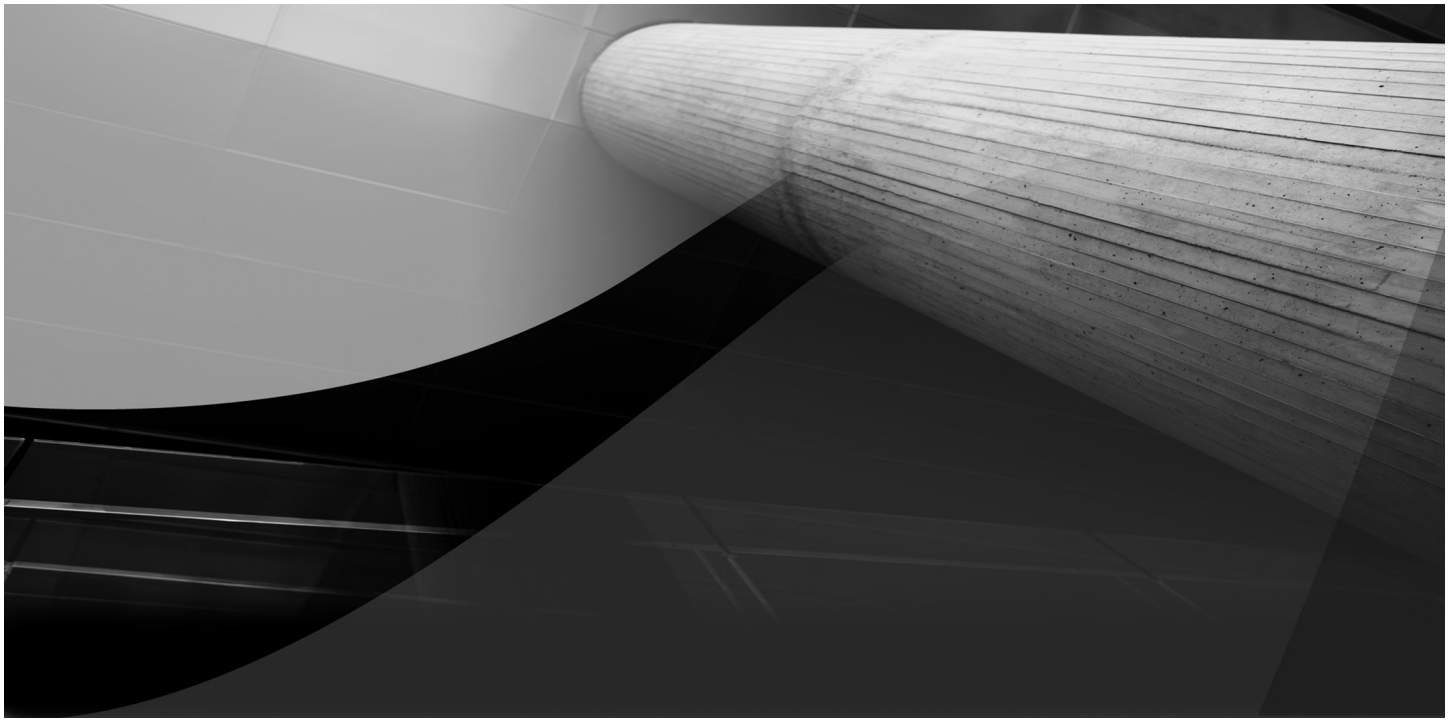
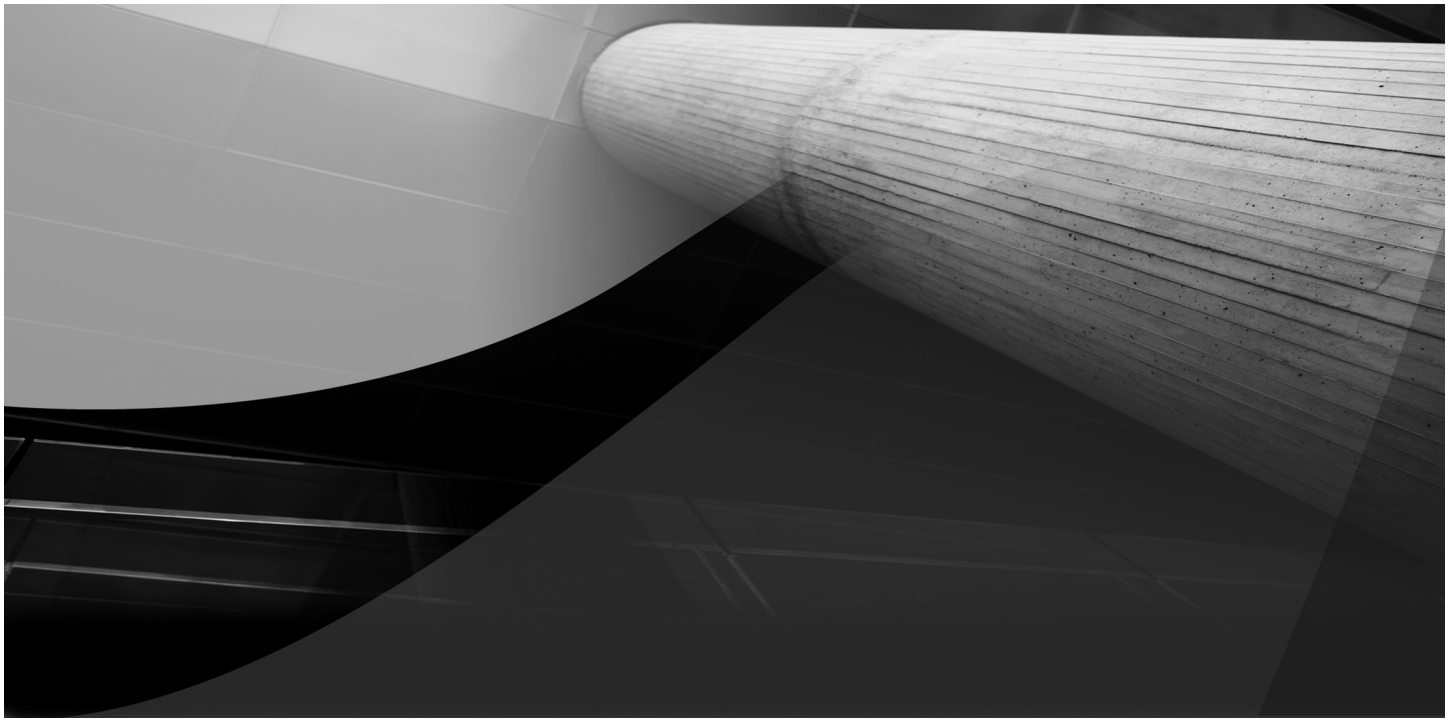


PART I

High Availability Architecture and Clusters



CHAPTER 1

Introduction to High Availability and Scalability

In today's fast-paced world, data and application *availability* can make or break a business. With access to these businesses granted via the ubiquitous and "always-on" Internet, data availability is an extremely important component in any business function.

Database systems are growing at an enormous rate in terms of the number of simultaneously connected and active users as well as the volume of data they handle. Even though the servers used to store huge, active databases have also improved in performance and capacity, a *single* server, powerful though it may be, is frequently unable to handle the database load and capacity requirements for these active databases. This factor makes it necessary to scale the processing capacity or scale the hardware or software to accommodate these requirements.

High Availability

When availability is crucial for a business, extremely high levels of disaster tolerance must allow the business to continue in the face of a calamity, without the end users or customers noticing any adverse consequences. The effects of global companies conducting business across time zones spanning "24 × 7 × forever" operations, e-commerce, and the challenges associated with today's "flat world" all-drive businesses to achieve a level of disaster tolerance capable of ensuring continuous survival and profitability.

Different businesses require different levels of risk with regard to loss of data and potential downtime. A variety of technical solutions can be used to provide varying levels of protection with respect to these business needs. The ideal solutions would have no downtime and allow no data to be lost. Although such solutions do exist, they are expensive, and hence their costs must be weighed against the potential impact of a disaster and its effects on the business.

Because computers are capable of working at faster and faster rates, the businesses that depend on them are placing more and more demands on them. As a result, the various interconnections and dependencies in the computing fabric, consisting of different components and technologies, are becoming more complex every day. The availability of worldwide access via the Internet is placing extremely high demands on businesses as well as the IT departments and administrators that run and maintain these computers in the background.

Adding to this complexity is the globalization of businesses, which ensures that there is no "quiet time" or "out-of-office hours" so essential to the maintenance requirements of these computer systems. Hence, businesses' computer systems—the lifeblood of the organization—must be available at all times: day or night, weekday or weekend, local holiday or workday. The term *24 × 7 × forever* effectively describes business computer system availability and is so popular that this term is being used in everyday language to describe non-computer–based entities such as 9-1-1 call centers and other emergency services.

The dictionary defines the word *available* as follows:

1. Present and ready for use; at hand; accessible.
2. Capable of being gotten; obtainable.
3. Qualified and willing to be of service or assistance.

When applied to computer systems, the word's meaning is a combination of all these factors. Thus, access to an application should be present and ready for use, capable of being accessed, and qualified and willing to be of service. In other words, an application should be available

easily for use at any time and should perform at a level that is both acceptable and useful. Although this is a broad, sweeping statement, a lot of complexity and different factors come into play before true high availability is achieved and sustained.

HA Terminology

The term *high availability* (*HA*), when applied to computer systems, means that the application or service in question is available all the time, regardless of time of day, location, and other factors that can influence the availability of such an application. In general, it is the ability to continue a service for extremely long durations without any interruptions. Typical technologies for HA include redundant power supplies and fans for servers, RAID (Redundant Array of Inexpensive/ Independent Disks) configuration for disks, clusters for servers, multiple network interface cards, redundant routers for networks, and even multiple datacenters within the same metro area to provide an extremely high level of availability and load balancing.

Fault Tolerance

A *fault-tolerant* computer system or component is designed so that, in the event of component failure, a backup component or procedure can immediately take its place with no loss of service. Fault tolerance can be provided with software, embedded in hardware, or provided by some combination of the two. It goes one step further than HA to provide the highest possible availability within a single datacenter and within a single application execution environment such as a database.

Disaster Recovery

Disaster recovery (DR) is the ability to resume operations after a disaster—including destruction of an entire datacenter site and everything in it. In a typical DR scenario, significant time elapses before a datacenter can resume IT functions, and some amount of data typically needs to be reentered to bring the system data back up to date.

Disaster Tolerance

The term *disaster tolerance (DT)* is the art and science of preparing for disasters so that a business is able to continue operation after a disaster. The term is sometimes used incorrectly in the industry, particularly by vendors who can't really achieve it. Disaster tolerance is much more difficult to achieve than DR because it involves designing systems that enable a business to continue in the face of a disaster, without the end users or customers noticing any adverse effects. The ideal DT solution would result in no downtime and no lost data, even during a disaster. Such solutions do exist, but they cost more than solutions that have some amount of downtime or data loss associated with a disaster.

Planned and Unplanned Outages

So what happens when an application stops working or stops behaving as expected, due to the failure of even one of the crucial components? Such an application is deemed *down* and the event is called an *outage*. This outage can be planned for—for example, consider the outage that occurs when a component is being upgraded or worked on for maintenance reasons.

Whereas planned outages are a necessary evil, an unplanned outage can be a nightmare for a business. Depending on the business in question and the duration of the downtime, an unplanned outage can result in such overwhelming losses that the business is forced to close. Regardless of the nature, outages are something that businesses usually do not tolerate. There is always pressure

on IT to eliminate unplanned downtime totally and to drastically reduce, if not eliminate, planned downtime. We will see later how these two requirements can be effectively met for at least the Oracle database component.

Note that an application or computer system does not have to be totally down for an outage to occur. It is possible that the performance of an application degrades to such a degree that it is unusable. In this case, although the application is accessible, it does not meet the third and final qualification of being willing to serve in an adequately acceptable fashion. As far as the business or end user is concerned, this application is down, although it is available. We will see later in this book how Oracle Real Application Clusters (RAC) can provide the horizontal scalability that can significantly reduce the risk of an application not providing adequate performance.

An End-to-End Perspective

From the start, you should be clear that high availability is not just dependent on the availability of physical components such as hardware, system software (operating system and database), environment, network, and application software. It is also dependent on other "soft" resources such as experienced and capable administrators (system, network, database, and application specialists), programmers, users, and even known, repeatable business processes.

It is entirely possible that a business installs and configures highly available "hard" components but does not employ competent administrators who are able to maintain these systems properly. Even if the administrators are competent, availability can be adversely affected when a business process, such as change control, is not followed properly, and incorrect, untested changes are made that could bring such a system down. High availability therefore needs to be seen with an end-to-end perspective that covers all aspects.

Having said this, we should now define the *single point of failure (SPOF)*—any single component that can bring down the entire system as a result of failure. For example, in a computer system that has a single controller interfacing with the disk subsystem, a hardware failure of this controller will bring the whole system down. Although the other components are working, this one *single* component has caused a failure. Identification of and protection against SPOFs are crucial tasks of providing HA.

It is not possible to cover all aspects of HA in an Oracle-specific book such as this. We will cover only how HA can be achieved specifically in the area of the Oracle RDBMS, which is an important component of the HA picture. We will also equip you—the database administrator, programmer, or architect—with techniques that will enable you to achieve HA in this area.

What's more, HA is not something that can be achieved simply by installing HA-aware hardware and software components, employing competent administrators, creating proper procedures, and walking away from it all. The HA process needs continual adjustment, evolution, and adaptation to changing circumstances and environments. Also, this uphill battle occurs on a continual basis—so be prepared!

Cost of Downtime

As hinted at earlier, there is a cost to downtime, just as there is a cost to ensuring that downtime is drastically reduced or even completely eliminated. The trick is to build your systems so that they never go down, even though you *know* that they *will* go down at some time or another. Making downtime the last option will ensure HA. Of course, most companies cannot continue to throw large sums of money at this issue. At some point in time, the additional money spent will return only marginal benefits. Therefore, it is essential to price out your downtime and then use that

figure to determine how much you can afford to spend to protect against planned/unplanned downtime. With some effort and experience, this expense can be determined, and you might want to use this information while providing various options and scenarios to management.

The cost of being down usually amounts to lost user productivity, and the actual cost is mostly dependent on what work the users perform when accessing the affected systems. For example, if your development server went down for one hour during prime office time, and 10 developers sat idle for that hour waiting for the server to come up, and each developer costs $100 per hour, then the downtime has effectively cost $100 × 10 × 1 = $1,000. However, if the server that went down served a major shopping site on the Internet during a holiday gift-buying season, you might count the losses in millions of dollars, even if the downtime was brief, because shoppers may move away to a competing site rather than wait for yours to become usable. Figure 1-1 shows a sample chart comparing downtime to cost.

The potential cost of downtime is also dependent on various factors such as time of day and duration of the downtime. For example, an online stock brokerage firm cannot afford to be down even for seconds during business hours. On the other hand, it could go down for hours during nontrading hours without any consequences. Cost of downtime is not linearly dependent on the duration of the downtime. For example, a two-hour outage may not necessarily cost the same as two one-hour downtime periods.

One helpful trick used with balancing the cost of downtime versus the cost of ensuring against downtime is the "availability curve." The more you spend on HA components, the higher you move up the curve. However, the incremental costs of moving from one level to the next increase as you move up the curve.

Here are the four distinct levels of system availability components on the curve:

- **Basic systems** These are systems with no protection or those that employ no special measures to protect their data and accessibility. Normal tape backups occur at scheduled intervals, and administrators work to restore the system from the last known good backup if and when it breaks. There is no extra cost for HA.

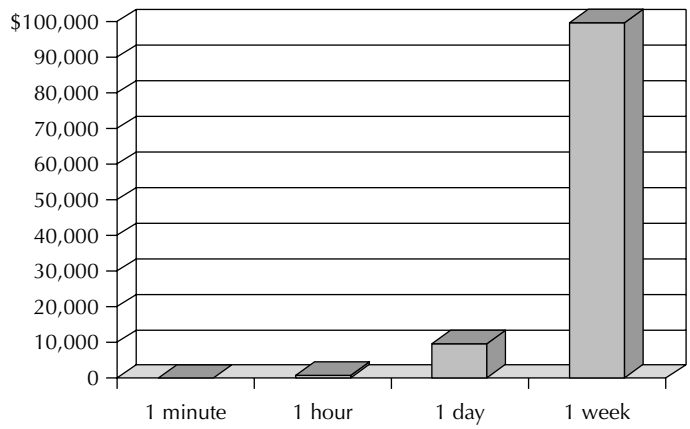


FIGURE 1-1 *Cost of downtime*

- **Redundant data** Some level of disk redundancy is built into the system to protect against loss of data due to disk failures. At the most basic level, this is provided by RAID 5 or RAID 1–based disk subsystems. At the other end of the scale, redundancy is provided by storage area networks (SANs) that have built-in disk-protecting mechanisms such as various RAID levels, hot-swappable disks, "phone home"-type maintenance, and multiple paths to the SAN. The cost of such protection includes procurement of the SAN, attendant SAN fabric and controllers, as well as extra sets of disks to provide RAID protection.
- **System failover** In this case, two or more systems are employed to do the work of one. When the primary system fails, the other, usually called the "secondary" system, takes over and performs the work of the primary. A brief loss of service occurs, but everything quickly works as it did before the failure. The cost of this solution is more than double that of basic systems. Usually, a SAN needs to be employed to make sure that the disks are protected and to provide multiple paths to the disks from these servers.
- **Disaster recovery** In this case, in addition to the systems at the main site (which in themselves may incorporate the previous highest level of protection), all or part of these systems are duplicated at a backup site that is usually physically remote from the main site. You must develop ways of replicating the data and keeping it up to date. The costs are more than double that of the previous level, because you will also have to duplicate an entire site, including datacenters, real-estate facilities, and so on.

As you can easily see, higher and higher levels of availability equate to escalating costs. When faced with even a rough estimate of cost, business leaders (and especially accounting staff) are quick to adjust their levels of expectancy.

Underpinning every aspect, of course, is the fact that you are monitoring, measuring, and recording all this uptime (or downtime, as the case may be). It is a given that you cannot quantify what you do not measure. Nevertheless, many organizations that demand 100-percent uptime do not even have basic measurement tools in place.

Five Nines

The phrase "five nines" is usually thrown about during discussions of high availability, and you need to understand what this means before agreeing (as an administrator or system architect) to provide such a level of availability. A user or project leader will invariably say that 100-percent availability is a necessity, and barring that, at least five nines availability must be maintained—that is, 99.999-percent availability.

To make this concept a bit clearer, Table 1-1 compares uptime and downtime percentages to real-time figures. As you study this table, keep in mind that the cost of providing higher and higher levels of uptime becomes progressively (and sometimes prohibitively) expensive. As you work with management, understanding this can help you provide a clear explanation of these terms and what they mean when translated to actual downtime and attendant costs.

Building Redundant Components

High availability is made possible by providing availability in multiple layers of the technical stack. The inclusion of redundant components that reduce or eliminate SPOFs is the primary key in achieving high availability. For example, more than one host bus adaptor (HBA), a controller for communicating with remote disks, is usually present in each server that connects to a SAN.

Percent Uptime	Percent Downtime	Downtime per Year	Downtime per Week
98	2	7.3 days	3 hours, 22 minutes
99	1	3.65 days	1 hour, 41 minutes
99.8	0.2	17 hours, 30 minutes	20 minutes, 10 seconds
99.9	0.1	8 hours, 45 minutes	10 minutes, 5 seconds
99.99	0.01	52.5 minutes	1 minute
99.999	0.001	5.25 minutes	6 seconds

TABLE 1-1 *Uptime Percentage with Real-Time Figures*

These HBAs, in turn, are able to connect into two or more network adaptor switches to which the SANs are themselves connected. This way, the failure of one HBA or even one network switch will not bring down the server and the application hosted on that server. *Multihosting* (the ability to attach multiple hosts to a single set of disks) and *multipathing* (the ability to attach a single host to its set of disks via more than one path) are common ways of introducing redundancy in such HA systems.

Redundant components exist in the software layers as well. For example, multiple web servers can be front-ended by a load balancer that directs all web requests to a bank of web servers. In this case, when one web server fails, existing connections migrate over to surviving web servers, and the load balancer connects new requests to these surviving web servers.

Redundancy is not restricted to hardware and software, however. Redundancy also includes building physical, environmental, and other elements into the framework. Most of the major Internet datacenters or Internet exchange points now have complete redundancy in terms of power, air conditioning, and other factors, so that the failure in any one of the provider's resources won't affect the operation.

In New York City, for example, two telecommunication systems were strategically placed in the erstwhile World Trade Center complex—one in each tower—with the assumption that the probability of both buildings collapsing was close to zero. However, unfortunately, that assumption was proved wrong. Now, companies are building redundant datacenters that are geographically separated across state or even country boundaries to avoid natural or other catastrophic events. Availability of dark fibers and the improvements in technology such as dense wavelength division multiplexers (DWDMs) make this possible.

Redundancy in the network layer is achieved through the redundant hardware engines in a chassis, a redundant network through multiple chassis, or a combination of the two. Host protocols such as ICMP Route Discovery Protocol (IRDP), Cisco's Hot Standby Routing Protocol (HSRP), and Virtual Router Redundancy Protocol (VRRP) help choose the best next-hop router to reach if one of the routers is unavailable from the server's perspective. In the routing level, Non-Stop Forwarding (NSF) protocol suites combined with millisecond timers reduce the failure or switchover time in case of primary hardware switching engine failure.

In the transport level, physical layer redundancy can be achieved by SDH/SONET self-healing, which restores the traffic in an alternate path in case of fiber link failure. During early 2000, a major transport provider experienced a fiber cut in its long-haul, coast-to-coast transport

network in the United States and rerouted the traffic through Europe without most of the end users knowing that the rerouting even took place.

Also, it is now possible to provide redundant database services via the Oracle RAC, and you will see this in detail in subsequent chapters. Suffice it to say at this time that redundancy in database services is an important part of providing HA in the organization, and Oracle RAC enables such a provision.

Of course, adding redundancy into the system also increases its cost and complexity. We hope that the information contained in this book can help you understand that complexity and ease your fears about managing such a complex environment.

Common Solutions for HA

Depending on your budget, you can arrive at a number of solutions for providing high availability. Clustering servers has been a common way to build a highly available and scalable solution. You can provide increasing levels of HA by adopting one of the higher levels of protection described earlier. In most current datacenters, RAID disks, usually in SANs, provide at least a basic level of disk protection. Failover servers at the third level provide some protection from server failure. At the highest level, the disaster recovery site protects against drastic site failure.

Oracle technology can be used to provide all these levels of protection. For example, you can use Automatic Storage Management (ASM) to provide protection at the disk level, Oracle RAC to provide failover protection at the database level (in addition to database-level load balancing), and Oracle standby and Oracle replication to provide site protection failure. Of course, all this requires varying levels of support at the hardware, network, and software layers.

Cluster, Cold Failover, and Hot Failover

Although we will be dealing with clustering in detail in subsequent chapters, we will define it here. A *cluster* is a set of two or more similar servers that are closely connected to one another and usually share the same set of disks. The theory is that in the case of failure of one of the servers, the other surviving server (or servers) can take up the work of the failed server. These servers are physically located close to one another and connected via a "heartbeat" system. In other words, they check one another's heartbeats or live presence at closely defined intervals and are able to detect whether the other node is "dead" within a short period of time. When one of the nodes is deemed nonresponsive to a number of parameters, a failover event is initiated and the service of the nonresponsive node is taken over by other node(s). Additional software may also allow a quick takeover of one another's functions.

Clusters can be implemented in many configurations. When one or more servers in a cluster sit idle, and takeover from another server (or servers) occurs only in the case of a failure, a *cold failover* occurs. When all servers in a cluster are working, and the load is taken on by the surviving server (or servers), this is called a *hot failover*. Assuming that all the servers in the cluster are similar in configuration, in a cold failover, the load carried by the surviving server is the same. In case of a hot failover, however, the load taken on by the surviving server may be more than it can handle, and thus you will need to design both the servers and the load carefully.

There are three general approaches to system failover. In order of increasing availability, they are *no failover, cold failover,* and *hot failover*. Each strategy has a varying recovery time, expense, and user impact, as outlined in Table 1-2.

Variations on these strategies do exist: For example, many large enterprise clients have implemented hot failover and also use cold failover for disaster recovery. It is important to

Approach	Recovery Time	Expense	User Impact
No failover	Unpredictable	No to low cost	High
Cold failover	Minutes	Moderate	Moderate
Hot failover	Immediate or in seconds	Moderate to high	None*

TABLE 1-2 *Failover Approach and Impacts*

differentiate between failover and disaster recovery. *Failover* is a methodology used to resume system availability in an acceptable period of time, whereas *disaster recovery* is a methodology used to resume system availability when all failover strategies have failed.

No Failover

If a production system fails due to a hardware failure, the database and application are generally unaffected. Disk corruption and disk failures, of course, are an exception. Therefore, disk redundancy and good backup procedures are vital to mitigate problems arising from disk failure.

With no failover strategy in place, system failures can result in significant downtime, depending on the cause and your ability to isolate and resolve them. If a CPU has failed, you replace it and restart, while application users wait for the system to become available. For many applications that are not business critical, this risk may be acceptable.

Cold Failover

A common and often inexpensive approach to recovery after failure is to maintain a standby system to assume the production workload in the event of a production system failure. A typical configuration has two identical computers with shared access to a remote disk subsystem.

After a failure, the standby system takes over the applications formerly running on the failed system. In a cold failover, the standby system senses a heartbeat from the production system on a frequent and regular basis. If the heartbeat consistently stops for a period of time, the standby system assumes the IP address and the disk formerly associated with the failed system. The standby can then run any applications that were on the failed system. In this scenario, when the standby system takes over the application, it executes a preconfigured start script to bring the databases online. Users can then reconnect to the databases that are now running on the standby server.

Customers generally configure the failover server to mirror the main server with an identical CPU and memory capacity to sustain production workloads for an extended period of time. Figure 1-2 depicts server connections before and after a failover.

Hot Failover

The hot failover approach can be complicated and expensive, but it comes closest to ensuring 100-percent uptime. It requires the same degree of failover used for a cold failover but also

* To be precise, saying there is no user impact in a hot failover scenario is inaccurate. Very few systems are truly "hot" to the point of no user impact; most are somewhat "lukewarm," with a transient brownout.

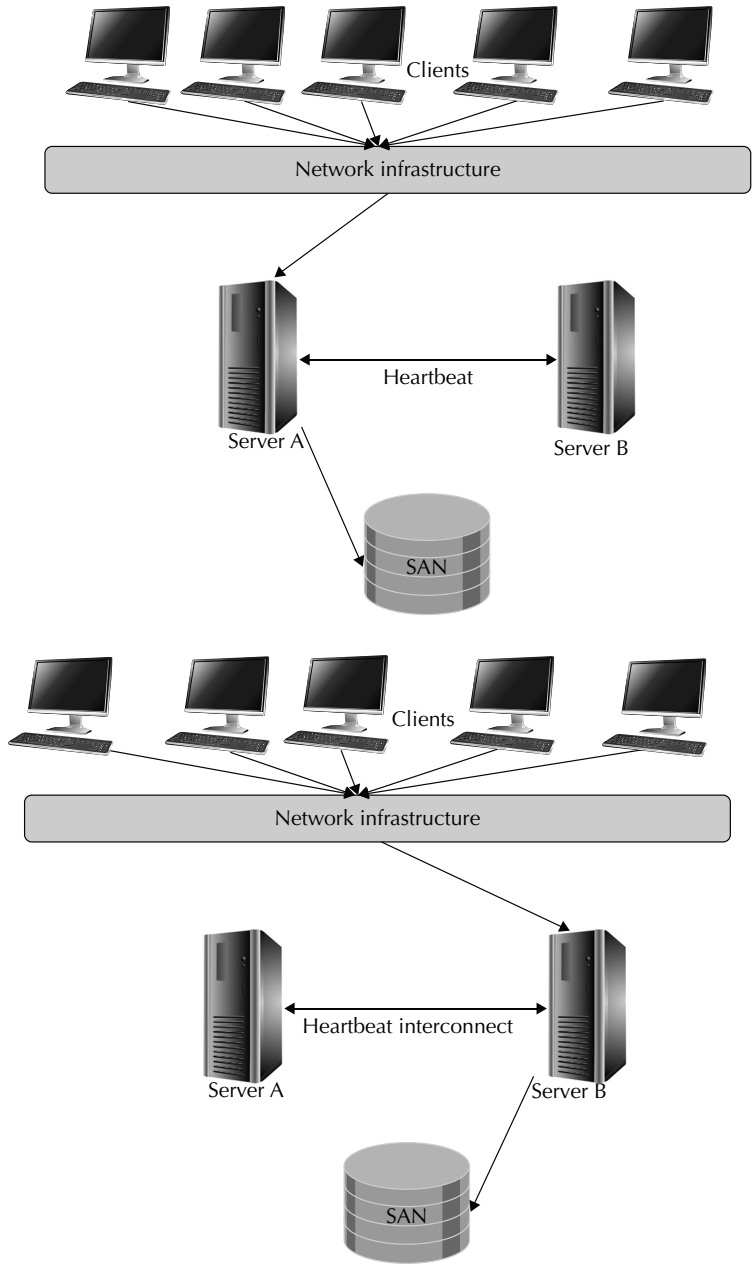


FIGURE 1-2 *Server connections before (top) and after (bottom) a failover*

requires that the state of a running user process be preserved to allow the process to resume on a failover server. One approach, for example, uses a three-tiered configuration of clients and servers. Hot failover clusters are normally capable of client load balancing. Oracle RAC supports hot failover configuration by transparently routing the incoming connections to the services in surviving nodes.

Table 1-3 shows load distribution of a 3,000-user workload in a three-node cluster. During normal operation, all nodes share approximately an equal number of connections; after failover, the workload from the failed node will be distributed to surviving nodes.

The 1,000 users on servers A and C are unaware of server B's failure, but the 1,000 users who were on the failed server are affected. However, systems A, B, and C should be appropriately configured to handle the additional load during unexpected node failures. This is one of the key elements during the capacity planning for clusters.

Table 1-4 summarizes the most common aspects of cold failover versus hot failover.

State	A	B	C
Normal	1,000 users	1,000 users	1,000 users
B fails	1,000 users	0 users	1,000 users
B users log on again	1,500 users	0 users	1,500 users

TABLE 1-3 *Workload Distribution During Cluster Failovers*

Aspects	Cold Failover	Hot Failover
Scalability/number of nodes	Scalability limited to the capacity of the single node.	Because nodes can be added on demand, it provides infinite scalability. High number of nodes supported.
User interruption required	Required up to a minimal extent. The failover operation can be scripted or automated to a certain extent.	Not required. Failover is automatic.
Transparent failover of applications	Not possible.	Transparent application failover will be available where the sessions can be transferred to another node without user interruption.
Load balancing	Not possible; only one server will be used.	Incoming load can be balanced between both nodes.
Usage of resources	Only one server at a time; the other server will be kept idle.	Both the servers will be used.
Failover time	More than minutes because the other system must be cold started.	Less than a minute; typically within a few seconds.

TABLE 1-4 *Cold Failover vs. Hot Failover*

HA Option Pros and Cons

Each HA option has its own advantages and disadvantages. Costs of setup and running the service are important to consider when deciding which HA option to use. At the end of the day, as an administrator or system architect, you are responsible for costing out the various options and helping management decide what is best. What's more, you will need to figure in the additional complexity of maintaining various configurations, remembering that as you add more redundancy into the system, you are also increasing the options for failure when handling these now complex configurations. In addition, employing consultants or engaging third-party vendor professional services to set up these complex configurations, deploy additional hardware and software, and maintain these systems can also quickly add to the basic costs.

Scalability

As mentioned at the beginning of the chapter, even powerful servers cannot always handle database load and capacity requirements. Server scalability can be improved using one or more of the following methods:

- Increase the processor count on the system, or *scale up* the computing resources.
- Increase the amount of work done in a given time via application tuning or *speed up* the processing.

The most common view of scaling is that of hardware scaling, which has at least as much to do with the software components as with the hardware. But what do you do when you cannot increase the processor count because you have reached the maximum capacity for that line of servers, or when you have tuned all the workloads as best you can and no more tuning opportunities exist?

Initial solutions to these problems include the use of multiple application copies and databases, but these result in data-sync problems and other process issues. The best solution, of course, is the use of clustered servers that can collectively perform much better than a single server for many applications. In other words, we can use clusters of servers to *scale out* (also known as *horizontal scalability)* rather than *scale up* (also known as *vertical scalability)*. It is in the provision of horizontal scalability where Oracle Real Application Clusters (RAC) excels.

Oracle Real Application Clusters Solution

Oracle Corporation introduced database clustering with Oracle version 6.2 exclusively on the DEC VAX/VMS. We will deal with many details of Oracle RAC in later chapters and see how it provides for high availability and scalability.

Essentially, Oracle RAC provides the ability for multiple servers to consistently access a single copy of the data. Theoretically, as the requirement to access this single copy increases, you keep adding nodes to the cluster. This ability to provide consistent access is not simple—the process requires a lot of coordination between the various nodes in the cluster. Oracle RAC does this efficiently, and we will see how exactly this occurs in later chapters.

Although Oracle RAC scales well, there is an upper limit on horizontal scalability. In general, application scalability is based on how good the application works in a single instance. If the SQL statements executed by the application are efficient and use an expected and reasonable amount

of resources (usually measured by Logical I/O and/or Physical I/O counts), you can generally expect this to scale well. In other words, you might compare Oracle RAC to a stereo amplifier: If the quality of the recording, whether on an audio tape or a digital device, is bad, placing even the best amplifier in front of it will not solve the problem. Instead, it will amplify the problem and make the situation unpleasant. This is also applicable for Oracle RAC or any other scalability solution. Hence, you will need to make sure application-level tuning is performed to remove bottlenecks before using clustering to scale out.

With the constant downward pressure on improving Total Cost of Ownership (TCO), businesses have chosen to move away from "Big Iron," monolithic servers to smaller sets of lower-cost "commodity" servers, And this is where Oracle RAC has truly come into its element because it helps businesses realize this paradigm shift by enabling large workloads to run on clusters of lower-cost servers rather than single, monolithic boxes. Also, such servers are able to scale up or down to the workload easily. Many new features in Oracle Database 11*g* RAC, such as server pools and SCAN (Single Client Access Names listeners, discussed in later chapters) provide the ability to perform this scaling seamlessly without interruption to the business.

Along with near-linear scalability, Oracle RAC–based systems can be configured to eliminate SPOF as far as the database layer is concerned. When database servers fail, applications based on Oracle RAC systems simply keep running. When designed and coded properly, this application failover is mostly transparent to users.

When combined with Oracle Data Guard, Oracle RAC is protected from major site failures. Oracle RAC enables horizontal scalability and thus the ability to support large, global, single-instance computing that hosts thousands of users. When protected via various HA options, such single global instances significantly reduce costs via consolidation in terms of servers, datacenters, software licenses, and skilled staff to maintain them.

Emerging Trends

Businesses today need to not just scale up but also to scale back—and to perform such scale-ups and scale-downs quickly, in a matter of a few hours or even minutes. This has placed an enormous demand on IT and datacenters to provide infrastructure not in a matter of days or weeks, but in terms of minutes. In other words, IT organizations should be able to commission and decommission computing services on the fly—something that was a pipe dream a few years ago.

IT organizations and vendors are now able to provide this scaling quickly and easily using a combination of first "virtualizing" computing resources and then building the ability to expose these resources in a metered and controlled fashion (namely "cloud computing"). This is achieved first by carving out virtual machines (VMs) from physical servers and presenting them as a service to both internal and external consumers.

IT vendors today are able to virtualize environments using products such as Oracle Virtual Servers (a hypervisor based on open-source Xen technology) to spin up computing resources on demand. Prime examples of the ability of vendors to provide cloud computing includes Amazon's Elastic Cloud Computing (EC2) and SalesForce.com's Sales Cloud 2. The former has quickly become sophisticated to the extent that they are now even able to provision complete Oracle E-Business Suite environments in minutes.

On the backend database side, this means that Oracle technologies should be able to scale database services as well. Oracle RAC plays a key role here because it provides scalability. However, the challenge is to perform this scaling dynamically without any interruption to the availability.

Oracle 11*g* Solutions

Oracle Database 11*g* RAC takes this challenge head on. The key requirement for dynamic provisioning is that the technology should be able to support resource movement and reassignments easily and dynamically from a pool, along with support for such functionality at all layers and components. We will dive into more detail in later chapters, but briefly, Oracle ASM provides the total abstraction at the storage layer, allowing multiple hosts to see not just database storage but share disk files as well, and dynamically adjust them to changing requirements. Oracle ASM also provides a complete suite of fully functioning Dynamic Volume Manager and File System for Oracle storage needs.

Also, server pools in the Oracle Grid Infrastructure foundation provide the capability of dynamically allocating resources within a Grid of Oracle RAC environment, thus providing flexibility at the database layer. SCAN IP provides a way to access a cluster using a single IP address, thus simplifying naming and administration.

In the latest version (namely Oracle Database 11*g*R2), Oracle provides the ability to spin up single, non–Oracle RAC instances for smaller loads and yet provide the high availability and "hot failover" to another node using RAC One. Edition-based Redefinition completes the high availability scenario because this feature can be used to provide application transparency during software changes as well—that is, online hot patching, the holy grail of downtime optimizations.

In a Nutshell

Modern business requirements have great impact on database and application availability, and vice versa. With ever-growing requirements and extreme dependence on information availability, the information systems are expected to remain fully functional and survive all external failures. The key to designing highly available systems relies on eliminating single-point failures in all critical components.

Oracle technologies are always a leap ahead on the current trends, making sure the enterprise requirements are met. Current versions of Oracle Clustering and Oracle Grid Infrastructure components allow us to increase and shrink the capacity on demand seamlessly. Oracle ASM and ACFS completely virtualized the storage infrastructure for the datacenter and have components built in to support continuous availability and transparent scalability.

Clusters provide an enterprise with uninterrupted access to their business-critical information, enabling the nonstop functions of the business. Clusters can be configured with various failover modes, depending on the requirements of the business. When designed and implemented judiciously, clusters also provide infinite scalability to business applications.