Oracle Rdb Hot Standby Performance Test Results

Bill Gettys (bill.gettys@oracle.com), Principal Engineer, Oracle Corporation
August 15, 1999

Introduction

With the release of Rdb version 7.0, Oracle offered a powerful and economical new solution to the reliability problem that has confounded data processing professionals from the earliest days of computing. Because of disk drive failures, the data we store in “persistent” storage isn’t really persistent. Disasters beyond our control—fires, floods and explosions—can destroy whole computer systems. The impact of the reliability problem is severe. Organizations that suffer total loss of critical computer systems may *never* recover.

Rdb 7.0’s new technique is called Hot Standby. Hot Standby’s approach is simple. Whenever a normal application makes a change to the master copy of the database, Hot Standby immediately makes that change to a standby copy of the database as well. Every *transaction* is replicated, almost instantly. The technique has minimal impact on the performance of the master database because it uses the highly optimized after image journal mechanisms that are already in place to recover a database from backup tapes.

What are the benefits of Hot Standby? First, if your master database ever goes down, you have a standby database that is accurate up to the last transaction. Second, you can achieve this with almost no slowdown to your production database. Third, you gain a standby database that can handle reporting or other read-only transactions, removing this workload from the master database, and therefore increasing the production application’s throughput.

Hot Standby replication is completely independent of the application programs that use the database. No application changes of any kind are required to use Hot Standby.

This paper has two parts.

Part 1. Hot Standby Design and Operation, which tells you: Hot Standby’s advantages over previous approaches to system reliability; describes its processes and operations; and explains its commands.

Part 2. Hot Standby Performance Testing, which describes Hot Standby performance in extremely high-volume OLTP workloads—and concludes that you can be very confident that Hot Standby can protect your database from almost any imaginable disaster, no matter how great your performance demands.

Throughout, tips are given to help you optimize your Hot Standby implementation.

Part 1: Hot Standby Design and Operation

Historical Attempts to Achieve System Reliability

When computers became indispensable to business, people began to look for ways to improve their reliability. A number of techniques have been tried. All had serious disadvantages.

Data Protection Scheme	Disadvantage
Backup data to magnetic tape	• As data volume grows, recovery time becomes unacceptably long. • Some schemes lose data since last backup. • Removing backup medium to a safe, remote location extends recovery time. • Physical transportation of backup media can be risky. • There is no provision for replacing damaged hardware.
Failover computer site	• Contracting for access to these sites can be expensive • Time to recover is long.
Fail-safe computer systems	• Purchasing these systems can be expensive. • Hardware is duplicated, with no benefit to normal processing.
Clusters (with disk shadowing)	• Elements of the clusters must be kept within a limited geographic range. • The unit of recovery is a disk drive, making network bandwidth expensive. • Time required to resume shadowing after a failure can be significant.
Traditional database replication	• Replication causes significant degradation of master database performance.

Table 1: Historical data integrity solutions

Hot Standby avoids all these problems:

- *Up to the second.* With Hot Standby, the standby database is kept up to date by applying transactions as they are committed on the master database. Since the database is continuously kept current, the only recovery time[1] required is the time needed to start the application on the standby system.

[1] Recovery time. In Hot Standby, the production software can't run on the standby database while it is in standby mode because read-write transactions aren't allowed. The recovery time is the time required to 1) decide that the master database is dead, 2) tell the standby database that it is no longer a standby database and now must accept read-write transactions, and 3) start the application on the standby database node. For

- *Guarantee: no data loss.* All data committed on the master database is immediately applied to the standby database. Options are available to manage the synchronization level between the two databases. For the most critical requirements, no committed data will be lost under any circumstances.
- *Choose the computer system size required for the standby system.* The standby computer system can be smaller than the master system if reduced performance is acceptable during a crisis
- *No geographic limit.* The distance between master and standby computer systems is limited only by the availability of network connections.
- *Low bandwidth requirements.* For most applications, the network bandwidth requirement for Hot Standby replication is remarkably low.
- *Fast recovery from network failure.* In other backup methods such as disk shadowing, recovering from network failure—and resulting loss of synchronization between master and standby databases—requires the copying of all disk drives, a task so formidable it can take many hours, even days. In Hot Standby, full recovery requires only copying the few transactions applied to the master database while the network was unavailable. As a result, catchup takes only minutes.
- *No performance 'drag'.* Hot Standby transactions consume almost no resources on the master database. As a result, the master database shows little or no performance degradation when using Hot Standby.

Process Model and Operation

Hot Standby is implemented using four extra processes, two on the master side and two on the standby side.

The master database uses these two processes.

- *ALS, or AIJ Logging Server*. This process is normally used to improve After Image Journal (AIJ) processing for busy databases. When Hot Standby is enabled, it takes on an additional task: shipping the AIJ blocks that will be written to the journal file to the standby side as well. In a VMS cluster, there is one ALS process per database per node.
- *LCS, or Log Catch-up Server* . This process determines whether the master and standby databases are configured properly for Hot Standby; ships any previously committed transactions to the standby database; and requests status messages from the standby database.

The standby database uses two process types as well.

- *AIJSERVER Process*. This process manages communication with master database processes. There is an AIJSERVER process for each master database process that needs to communicate with a process on the standby system.

a database in a mission-critical application, it is to be expected that step 2 will take only a few seconds, and step 3 will take only minutes.

- *LRS — Log Recovery Server* . This process applies changes from the master database to the standby database in much the same manner as journal records are applied during database recovery. There is a single LRS process per standby database.

There may also be an ABS Process.

If you have enabled automatic backup of database journals, an ABS process is created for the master database when a particular journal file is no longer needed for Hot Standby. The ABS process makes a backup copy of the journal and then clears the journal file for reuse. ABS processes on the standby database are *always* used to clear standby journal files when they are no longer needed, whether or not ABS is enabled for the master database.

Each of these processes creates log files detailing their operation. The name and location of the log files can be controlled using logical names.

Process	**Master Database**	**Standby Database**
ABS	RDM$BIND_ABS_LOG_FILE[2]	RDM$BIND_ABS_LOG_FILE
ALS	RDM$BIND_ALS_OUTPUT_FILE	
AIJSERVER		RDM$BIND_HOT_OUTPUT_FILE
LCS	RDM$BIND_LCS_OUTPUT_FILE	
LRS		RDM$BIND_LRS_OUTPUT_FILE

Table 2: Logical names for Hot Standby log files

> *Tip: Create log files for each Hot Standby process by defining the appropriate logical names. These log files ease diagnosis and repair of Hot Standby processes.*

Hot Standby Commands

Replication is managed and controlled using RMU/REPLICATE AFTER_JOURNAL[3] commands. The three major commands are:

- CONFIGURE. Set a database as either master or standby and set replication characteristics
- START. Start replication using characteristics set using the CONFIGURE command. This command must be issued on both master and standby nodes. Always issue the START command on the standby side first.
- STOP. Stops replication in an orderly fashion. You can stop replication by issuing this command for either the master or standby database. Replication stops more quickly if you issue this command for the master database.

[2] Hot Standby can be used for both Oracle Rdb and Oracle CODASYL DBMS databases. The equivalent logical names for DBMS databases begin with DBM$ instead of RDM$

[3] The equivalent command for Oracle CODASYL DBMS databases is DBO/REPLICATE AFTER_JOURNAL

While you can modify Hot Standby replication parameters using the START command, best practice is to set replication characteristics using only the CONFIGURE command.

> *Tip: Create command procedures for configuring the master database and the standby database exactly as they should be for production use. Always start Hot Standby by executing the configuration command procedure and then executing the RMU/REPLICATE AFTER START command. If you ever plan to operate in failback mode (original master database is now the standby database; old standby database is now the master database), create two additional configuration command procedures to reverse the replication characteristics. Incorporate these command procedures in the operational procedures that your computer system operators use to manage daily operations.*

The RMU/REPLICATE AFTER_JOURNAL REOPEN_LOG command can be used to create new versions of the ALS, LCS and LRS logs. In addition, new logical names have been added to keep these log files from growing too large. See the Rdb Version 7.0.2 Release Notes for details.

Database Synchronization

When Hot Standby Replication is first started, the LCS process determines whether the master and standby databases are able to participate in physical replication. The requirements that must be met are strict:

- The standby database must have been created by backing up the master database using RMU/BACKUP and restoring the backup file on the standby system.

- After Image Journals with Fast Commit optimization must be enabled. Best practice is to reserve some extra journal slots and to enable emergency journal creation.

- AIJ Log Server optimization must be set to automatic.

- Database open mode must be Manual.

You can use the Automatic Backup Server to backup journal files as they are no longer needed. You should always enable operator notification of database related events.

> *Tip: Find out whether a database is configured properly for Hot Standby by issuing this command:*

```
$ RMU/DUMP/HEADER=HOT <database-root-specification>
```

Synchronization Modes

As journal records are written to the master database After Image Journal, they are shipped to the standby database and stored in buffers there. (A temporary file is used to store change records for very large transactions.) When a commit record arrives at the standby database, that transaction's changes are applied to the standby database.

You can control the level of synchronization between the master and standby databases by selecting from the four available synchronization modes:

- *Cold***.** The master database does not wait for any confirmation from the standby database before telling master database users that a commit operation has successfully completed.
- *Warm***.** The master database does not confirm commits until the commit record has been received on the standby node.
- *Hot.* The master database confirms commits only when the transaction commit record has been written to the After Image Journal on the standby database.
- *Commit*. The master database confirms that a transaction has been committed only when the changes have been written to the standby database.

Cold synchronization has almost no impact on transaction completion times on the master database. Commit synchronization can cause significant delays in transaction completion time. The other modes fall between the extremes. Choosing the appropriate synchronization is one of the most important decisions to make for Hot Standby. Notice that you are not choosing a strategy that results in loss of data by choosing a colder synchronization level. If Hot Standby fails due to a network or other problem and is later restarted after the problem is corrected, the LCS process compares the master and standby journals and catches up any committed transactions missing from the standby database. If it is absolutely critical that no transactions have to be reentered in the event of failover to the standby database, you might choose Commit synchronization.

> *Tip: For most database systems, warm synchronization is sufficient. With warm synchronization, the commit record has been delivered to the standby node before the master database commit operation is confirmed. The transaction will be written in the standby database unless that node fails during commit processing. Only if the standby system fails and then the master system fails, and then you decide to failover to the standby system after it recovers, would a transaction that was committed on the master system fail to appear on the standby system.*

Replication Governor

With very heavy workloads, it is possible for the standby database LRS process to fall behind in processing journal records into buffers and committing transactions. To prevent the standby database from falling too far behind, the optional Replication Governor is used to dynamically increase the synchronization level from cold to warm to hot and eventually to Commit. While the Replication Governor is optional, it is highly recommended.

The Replication Governor automatically decreases the synchronization level to that specified by the database administrator when the workload is lower.

Tip: Always enable the Hot Standby Replication Governor.

Part 2: Hot Standby Performance Testing

A number of significant performance improvements have been added to Hot Standby in Rdb version 7.0.2. These have resulted in a greater than tenfold performance improvements in some benchmark tests. The critical question for Rdb customers is, “Can Hot Standby replicate my production database without impacting current performance levels?” Oracle has conducted a series of tests to determine the ultimate speed of Hot Standby replication and compare that with known customer workloads. The results of these tests are described below.

Online Transaction Processing Workload

Oracle created a 30 gigabyte database and executed an industry standard test workload similar to an order entry system for a product distribution business. In an attempt to generalize the results, the workload throughput was measured in units of AIJ growth. We measured the database throughput with Hot Standby disabled and with Hot Standby enabled. In addition, we measured the rate at which two hours of master database transactions could be applied to the standby database during initial Hot Standby synchronization. Workload measurements were made for a minimum of two hours.

The hardware used for these experiments was a pair of Compaq Alpha Servers with 8 processors and 8 gigabytes of memory. Disk storage was connected using four CI controllers and four star couplers for both the master and standby nodes.

We compared these results with AIJ growth rates achieved in a database believed to have the most intensive update workload in a production system. (If your workload is heavier, please contact the author.) The results of these experiments are shown in Figure 1 below:

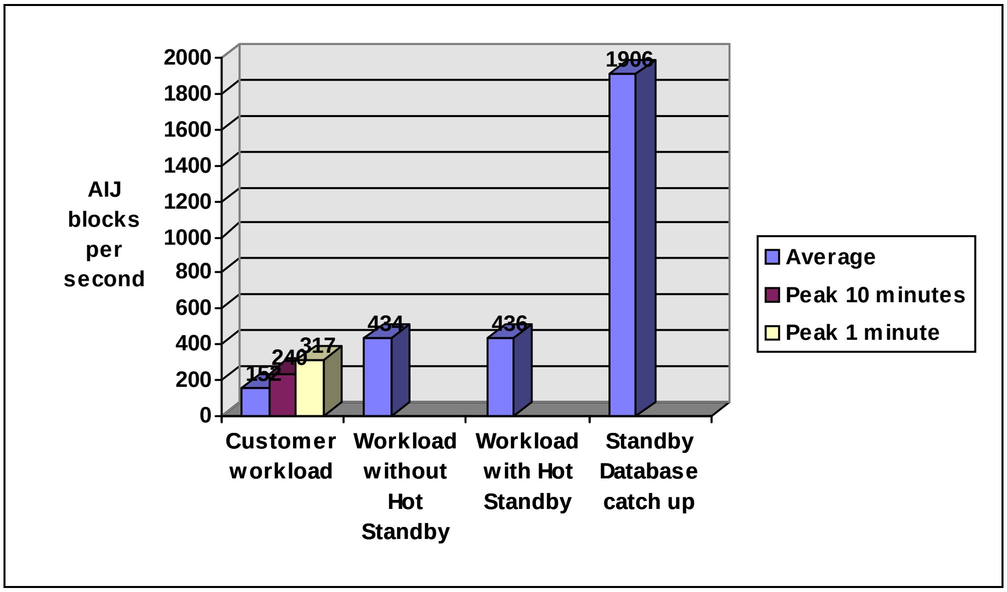

Figure 1: 30 gigabyte database and workload test results

Clearly, Hot Standby is capable of sustaining replication at AIJ growth rates significantly greater than those achieved by a large customer production system. The very high catch-up rate hints that this workload is much smaller than the maximum that Hot Standby can accommodate.

There is another striking observation from this experiment. Even with workloads exceeding the peak rate achieved by a very demanding customer application, Hot Standby imposes no measurable penalty on the master database. *In this test, the AIJ growth rate that was achieved with Hot Standby is essentially the same as that achieved without Hot Standby.*

Replication of Extreme Workloads

To determine the limit of Hot Standby's ability to replicate database changes, we built a new 150 gigabyte database on the same hardware. This time we used more partitions in the database, more server processes, and row cache to pin key tables and indexes in memory. The results of these tests are shown in Figure 2 below:

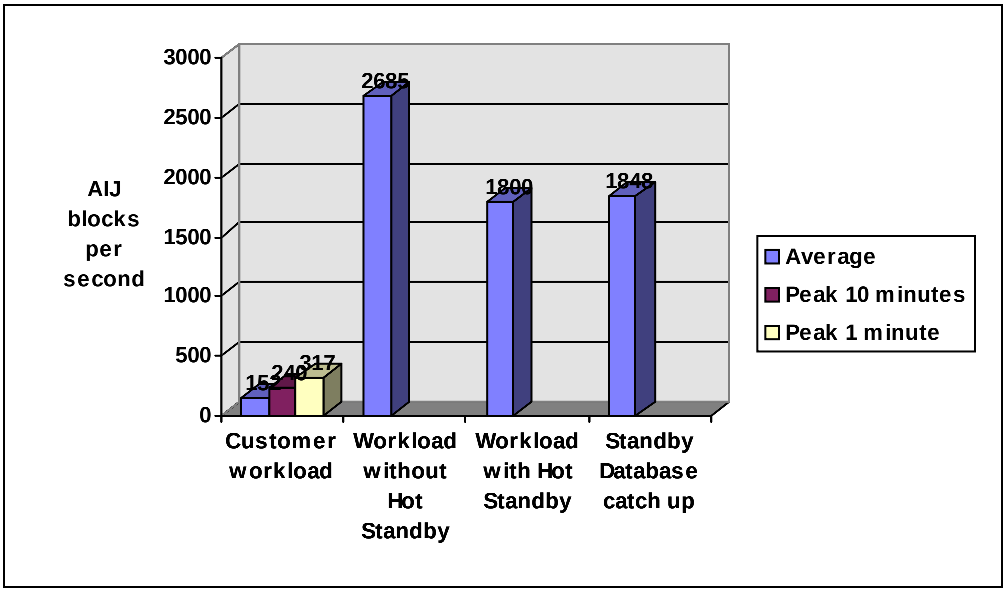

Figure 2: 150 gigabyte, Row Cache database and workload test results

This test shows that it is possible, using extraordinary means, to generate a workload that cannot be replicated by Hot Standby. This should cause little alarm for most customers, since Hot Standby can clearly replicate workloads that generate up to 1800 blocks of *sustained* AIJ growth per second, almost six times the peak one-minute rate achieved by the large customer workload.

It should be noted that near the Hot Standby replication limit of about 1800 blocks per second, the replication rate oscillates wildly between about 3000 blocks per second and 0 as the replication governor regulates master database throughput. This behavior is not likely to be acceptable in a production environment. The effective workload limit for Hot Standby is somewhat smaller than that measured in this experiment.

Effect of a Very Large Transaction on Hot Standby Replication

The workload used for these tests consists of five distinct transactions of varying size. All are small compared to some large batch workloads. To measure the effect of large transactions on the replication ability of Hot Standby, we repeated the workload but added an additional transaction that generated 340,000 blocks of AIJ file growth by itself. The effect of the large transaction is shown clearly in Figure 3 below:

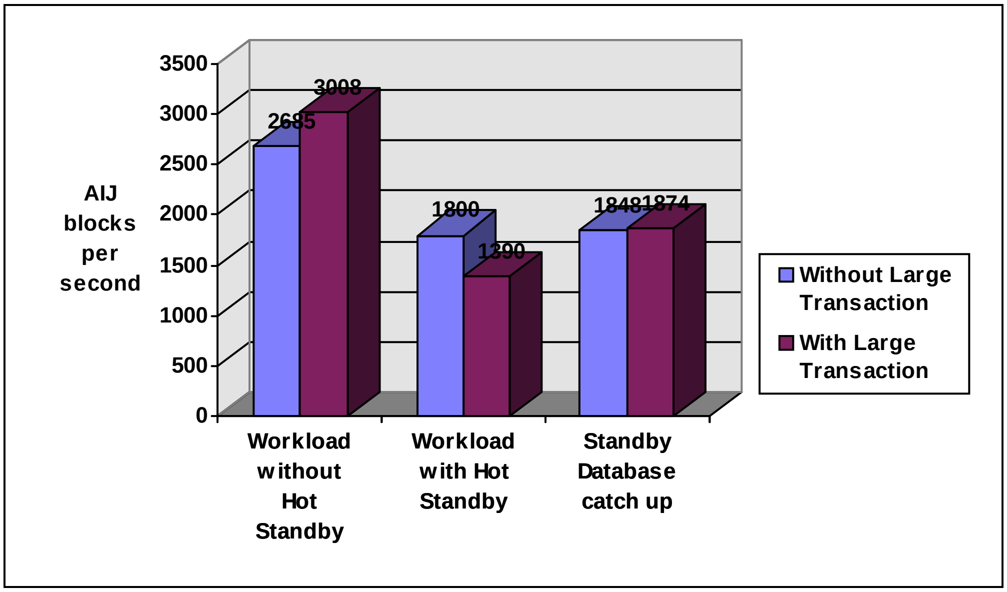

Figure 3: Effect of a large transaction on Hot Standby replication rate

As you would expect, adding an additional large transaction to the workload causes the total number of AIJ blocks written during the two-hour workload to increase by about 12%. The rate at which the standby database catch-up operation replicates changes to the master database remains about the same. There is a significant reduction in the amount of work that Hot Standby can replicate when it must process both an OLTP workload and an exceptionally large transaction.

Effect of Replicating Multiple Databases to a Single Standby Node

Customer experience has indicated that there can be a detrimental effect on replication throughput if several master databases, each on different master nodes, are replicated to a single standby node. To quantify this effect, we built three 30 gigabyte databases on each of three independent Alpha VMS systems and executed the same OLTP workload as in the first experiment on each of these systems. (Because of variations in the hardware used on each of these systems, the workload that could be achieved varied considerably from node to node.) We created standby databases for each of the three master databases on a single standby node. The results of these tests are shown in Figure 4 below:

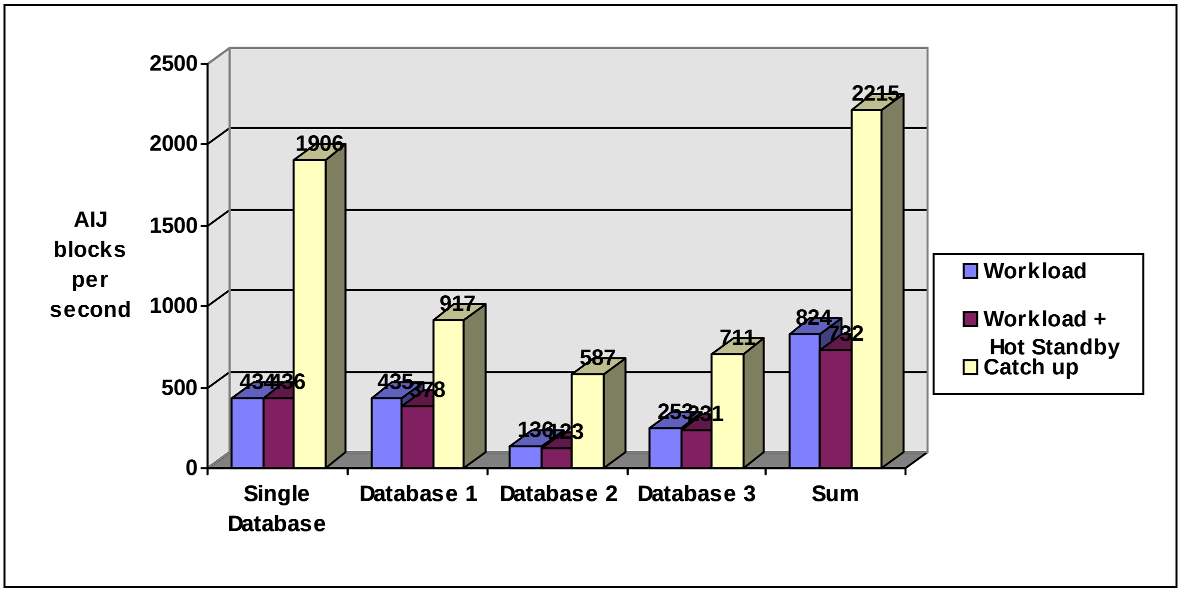

Figure 4: Effect of several master databases replicated to a single node

In this graph, the Single Database values represent the workload, replication, and catch-up rates measured in the experiment shown in Figure 1. In the new experiment, Database 1 was built on identical hardware to that used in the first experiment. As one would expect, the workload measurement was essentially identical. The additional workload on the standby node caused by the LRS and AIJSERVER processes of the other two databases caused a 13% reduction in the rate that the workload could be replicated and more than a 50% reduction in the catch-up rate for this database. The final group of columns indicates that the total amount of work completed on the standby node is greater when three databases are replicated. But the rate at which any single database can be replicated is reduced from the ideal case, in which there is no interference from other recovery processes.

It is reasonable to conclude that any other workload on the standby node, including generating reports from the standby database, will have a negative effect on the maximum rate at which databases can be replicated using Hot Standby.

> *Tip: If the workload that you must replicate using Hot Standby is large, avoid any activities on the standby database that could slow the AIJSERVER and LRS processes, slowing replication and eventually reducing the maximum workload achievable on the master database.*

Conclusions

Oracle Rdb’s Hot Standby feature has now been thoroughly tested for both performance and function in both customer and benchmark settings. The sustained rate at which Rdb can replicate databases is almost twenty times the average transaction rate achieved by the most active known Rdb customer database. You can be very confident that Oracle’s Hot Standby database replication feature can protect your database from almost any imaginable disaster, no matter how great the performance you require.