Database Mirroring & Snapshots
SQL Server 2008
Agenda – SQL Server User Group

- Database Mirroring – SQL Server 2008
- Database Mirroring 2005 vs 2008 benchmark
- Database Snapshots – SQL Server 2008
- SQL Performance & Tuning Tool (SQL Shot !)
Database Architechs

Paul Bertucci
- Founder Database Architechs
- Former Chief Data Architect – Veritas & Symantec
- **Currently Chief Architect - Autodesk**
- Author of SQL Server 2000, 2005 & 2008 Unleashed!
- Co-Author of ADO.NET in 24 hours
- Author MS SQL Server High Availability
- Author Sybase Performance & Tuning
- Author Sybase Physical DB Design
- Veritas SQL Server Performance Series
  pbertucci@dbarchitechs.com

Thierry Gerardin
- Managing Principal - Database Architechs
  - Management, Research and Development, DB Design and P&T for MS SQL Server, Sybase and Oracle
- European focus until recently/company growth leader – multiple years
- Expanding DB ARCH consulting practices to USA and Asia
- Sybase P&T and DBA Certified Professional
- Contributor to SQL Server 2005 and 2008 Unleashed!
  tgerardin@dbarchitechs.com
Database Architechs
Primary Services

- Data Analysis & Modeling
- Database Design
- Data Architecture
- Performance & Tuning
- High Availability
- Database Education & Training
- Distributed Data & Data Replication
- Master Data Management
- Business Intelligence & Data Delivery Platforms
<table>
<thead>
<tr>
<th>Past and Current Customers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel</td>
</tr>
<tr>
<td>Cisco</td>
</tr>
<tr>
<td>Charles Schwab</td>
</tr>
<tr>
<td>Aplia</td>
</tr>
<tr>
<td>AGIS</td>
</tr>
<tr>
<td>Veritas</td>
</tr>
<tr>
<td>Honda Motors</td>
</tr>
<tr>
<td>Toshiba Computers</td>
</tr>
<tr>
<td>Juniper Networks</td>
</tr>
<tr>
<td>Safeway Stores</td>
</tr>
<tr>
<td>Wells Fargo</td>
</tr>
<tr>
<td>Robert Half</td>
</tr>
<tr>
<td>Metalinc</td>
</tr>
<tr>
<td>Applied Materials</td>
</tr>
<tr>
<td>Apple Computers</td>
</tr>
<tr>
<td>PG&amp;E</td>
</tr>
<tr>
<td>Thomson Learning</td>
</tr>
<tr>
<td>Sybase</td>
</tr>
<tr>
<td>Symantec Corporation</td>
</tr>
<tr>
<td>Nissan Motors</td>
</tr>
<tr>
<td>Breg International</td>
</tr>
<tr>
<td>CSAA</td>
</tr>
<tr>
<td>Federal Express</td>
</tr>
<tr>
<td>Bank of America</td>
</tr>
<tr>
<td>Merrill Lynch</td>
</tr>
<tr>
<td>many others…</td>
</tr>
</tbody>
</table>
Database Mirroring?

A “complete” copy of a database that is created and maintained with as up to the second completeness that is possible → a mirror image.

What’s new or improved?
- Compression in log transactions across topology
- Various other performance enhancements
- Page fixing on the mirror side (AUTO)
- 3 years of production execution success!
Log Shipping

“Source”

SQL Server 2008

Primary Server

CallOne DB

translog

TxnLog

backups

\Backup\CallOne_tlog_200405141120.TRN

TxnLog

Copies

\LogShare\CallOne_tlog_200405141120.TRN

“Destination”

SQL Server 2008

Secondary Server

CallOne DB

TxnLog

Restores

\LogShare\CallOne_tlog_200405141120.TRN

“Monitor”

SQL Server 2008

Monitor Server

MSDB DB

Delay between
logs loaded

Delay Answer

Delay between
logs loaded

Delay Answer

Don’t build on this, it is being deprecated!
SQL Server 2008

Principal Server

Adventure Works DB

translog

Mirror Server

Adventure Works DB

translog

Witness Server

MSDB DB

Network

Client

Client

Client

Client

Copyright 2009 all rights reserved
Copy-on-write Technology

The new “copy-on-write” technology that Microsoft has created, is at the core of the database mirroring capability. A transaction from a client connection to the principal server (arrow label A) is written to the adventure works database (label D). Once the transaction is written to the principal servers transaction log, it is immediately copied (arrow label B) and written to the Mirror Server (also labeled D). When this physical log record is written to the mirror server, it sends back an acknowledgement (arrow label C) to the principal of its write success. This is the “copy-on-write” technology. The end result is that the mirror server is in the exact same state as the principal server (when the physical log record has been successfully written on the mirror side).
DB Mirroring Terms

The Principal database server - is the source of the mirroring. You can mirror one or more databases on a single SQL Server instance to another SQL Server instance. You cannot mirror a database on one SQL Server instance to itself (the same SQL Server instance).

The Mirror database server - will be the recipient of the mirroring from the principal database server. This mirrored database will be kept in a hot standby mode and cannot be used directly in any way. In fact, once you have configured database mirroring, this database will show its status as in continuous “restore mode”.

The Witness database server - is used when you want to be continuously checking to see if any failures have occurred to the primary database server and to help make the decision to failover to the mirror database server. It is optional, but a sound way to configure database mirroring. If you do not identify a witness server, the principal and mirror are left on their own to decide to fail-over or not.
DB Mirroring Modes

High-availability mode, High-protection mode, and High-performance mode.

[you must sacrifice levels of protection for performance]

Synchronous operation - a committed transaction will be committed (written) on both partners of the database mirroring pair. This obviously adds some latency cost to a complete transaction (it is across two servers). High-availability mode and High-protection mode use synchronous operations (termed High safety).

Asynchronous operation - transactions commit without waiting for the mirror server to write the log to disk. This can speed up performance significantly. High-performance mode uses asynchronous operations.

Whether the operations are asynchronous or synchronous depends on the “transaction safety” setting. This is controlled by the SAFETY option when configuring with Transact-SQL commands (Set to FULL for synchronous operations, set to OFF for asynchronous operations).
Fail-over methods

**Automatic failover** - is enabled with a three-server configuration; a principal, a mirror, and a witness server. Synchronous operations are required and the mirror database must already be synchronized (in sync with the transactions as they are being written to the principal). Role switching is done automatically. This is for **high-availability** mode.

**Manual failover** - is needed when there is no witness server and is also doing synchronous operations. The principal and the mirror are connected to each other and the mirror database must already be synchronized. Role switching is done manually. This is for **high-protection** mode.

**Forced Service** - in the case where there is a mirror server available, but it might not be synchronized. It can be forced to take over when the principal server has failed. This will possibly have data loss, since the transactions were not synchronized. This is for either **high-protection** or **high-performance** mode.
Rule-of-Thumb

You should use database mirroring:

- If you need to increase the availability of the database layer
- If you need to have automatic data protection (redundant storage of data)
- If you need to decrease the downtime that would normally be required to do upgrades

And very often:

- If you need to off load reporting (periodic data snapshots) without impacting the transactional system use database mirroring with database snapshots.
- Combo Needs: If you need data distribution, high availability and high data resiliency - use data replication with database mirroring.
Cannot be used with:

Database Mirroring cannot be used for any of SQL Server’s internal databases → TempDB, Master DB, MSDB, or Model DB.

Database Mirroring is NOT supported in SQL Server Workgroup Edition or Express Edition. **However**, these server editions could be used as Witness servers.

You cannot have database mirroring for databases enabled with FILESTREAM storage.
##Availability Continuum

<table>
<thead>
<tr>
<th>Availability Type</th>
<th>Characteristic</th>
<th>Availability Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extreme Availability</td>
<td>Near zero downtime!</td>
<td>(99.5% - 100%)</td>
</tr>
<tr>
<td>High Availability</td>
<td>Minimal downtime</td>
<td>(95% - 99.4%)</td>
</tr>
<tr>
<td>Standard Availability</td>
<td>With some downtime tolerance</td>
<td>(83% - 94%)</td>
</tr>
<tr>
<td>Acceptable Availability</td>
<td>Non-critical Applications</td>
<td>(70%-82%)</td>
</tr>
<tr>
<td>Marginal Availability</td>
<td>Non-production Applications</td>
<td>(up to 69%)</td>
</tr>
</tbody>
</table>

*Availability Range describes the percentage of time relative to the “planned” hours of operations*

8,760 hours/year | 168 hours/week | 24 hours/day
525,600 minutes/year | 7,200 minutes/week | 1,440 minutes/day
SQL Clustering basic configuration

Windows 2003 EE

SQL Server 2008 (physical)

SQL Server 2008 (Virtual SQL Server)

SQL Server 2008 (physical)

Windows 2003 EE

MSCS

Cluster Group Resources

Master DB
TempDB
Appl 1 DB

SQL Agent
SQL Full Text Search

Quorum Disk

MSCS

Copyright 2009 all rights reserved
Instance: **SQL08DE01**
Endpoint name: “EndPoint4DBMirroring1430”
Role: PARTNER

Instance: **SQL08DE02**
Endpoint name: “EndPoint4DBMirroring1440”
Role: PARTNER

Instance: **SQL08DE03**
Endpoint name: “EndPoint4DBMirroring1450”
Role: WITNESS
Database Properties - AdventureWorks2008

Ensure that security is configured for mirroring this database.

Server network addresses:
- Principal:
- Minor:
- Witness:

Note: Use fully qualified TCP addresses. For example: TCP://svr3.corp.abc.com:5022

Operating mode:
- High performance (asynchronous) - Commit changes at the principal and then transfer them to the minor.
- High safety without automatic failover (synchronous) - Always commit changes at both the principal and minor.
- High safety with automatic failover (synchronous) - Requires a witness server instance. Commit changes at both the principal and minor if both are available. The witness controls automatic failover to the minor if the principal becomes unavailable.

Status: This database has not been configured for mirroring
For SQL Server accounts in the same domain or trusted domains, specify the service accounts below. If the accounts are non-domain accounts or the accounts are in untrusted domains, leave the textboxes empty.

Service accounts for the following instances:

Principal:
ADS\bertucp

Witness:
ADS\bertucp

Complete the Wizard
Verify the choices made in the wizard and click Finish.

Click Finish to perform the following actions:

On the principal server instance, REM12374333\SQL108DE01
- Create the mirroring endpoint with the following properties:
  - Name: EndPoint4DBMirroring1430
  - Listener Port: 1430
  - Encryption: Yes
  - Role: Partner
  - Add the service account for the principal server instance, 'ADS\bertucp' as a login and grant it CONNECT permission on the mirroring endpoint.

On the mirror server instance, REM12374333\SQL108DE02
- Create the mirroring endpoint with the following properties:
  - Name: EndPoint4DBMirroring2440
  - Listener Port: 1440
  - Encryption: Yes
  - Role: Partner
  - Add the service account for the mirror server instance, 'ADS\bertucp' as a login and grant it CONNECT permission on the mirroring endpoint.

On the witness server instance, REM12374333\SQL108DE03
- Create the mirroring endpoint with the following properties:
  - Name: EndPoint4DBMirroring3450
  - Listener Port: 1450
  - Encryption: Yes
  - Role: Witness
  - Add the service account for the witness server instance, 'ADS\bertucp' as a login and grant it CONNECT permission on the mirroring endpoint.

After you specify the service accounts, logins will be created and will be granted CONNECT permission on the mirroring endpoint.

Do Not Start Mirroring

To use the specified network addresses for mirroring the database, click Start Mirroring. To wait to start mirroring, click Do Not Start Mirroring; you can then start mirroring by clicking Start Mirroring on the Mirroring page of the Database Properties dialog box. Alternatively, you can exit the Database Properties dialog box without starting mirroring now, but you will need to configure the operating modes and server network addresses again before you can start mirroring.
Creating Endpoints

-- create endpoint for PRINCIPAL server --
CREATE ENDPOINT [EndPoint4DBMirroring1430]
    STATE=STARTED
    AS TCP (LISTENER_PORT = 1430, LISTENER_IP = ALL)
    FOR DATA_MIRRORING (ROLE = PARTNER,
    AUTHENTICATION = WINDOWS NEGOTIATE
    , ENCRYPTION = REQUIRED ALGORITHM RC4)

SQL Server Log entry
01/05/2009 13:17:23,spid55,Unknown,The Database Mirroring protocol transport is now listening for connections.
01/05/2009 13:17:23,spid55,Unknown,Server is listening on [ 'any' <ipv4> 1430].
Creating Endpoints

-- create endpoint for MIRROR server --
CREATE ENDPOINT [EndPoint4DBMirroring1440]
STATE=STARTED
AS TCP (LISTENER_PORT = 1440, LISTENER_IP = ALL)
FOR DATA_MIRRORING (ROLE = PARTNER,
  AUTHENTICATION = WINDOWS NEGOTIATE
  , ENCRYPTION = REQUIRED ALGORITHM RC4)

-- create endpoint for WITNESS server --
CREATE ENDPOINT [EndPoint4DBMirroring1450]
  STATE=STARTED
  AS TCP (LISTENER_PORT = 1450, LISTENER_IP = ALL)
  FOR DATA_MIRRORING (ROLE = WITNESS,
  AUTHENTICATION = WINDOWS NEGOTIATE
  , ENCRYPTION = REQUIRED ALGORITHM RC4)
Backup Principal

- BACKUP DATABASE [AdventureWorks]
  - TO DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL10.SQLO8DE01\MSSQL\Backup\AdventureWorks4Mirror.bak'
  - WITH FORMAT
- GO
  - -- BACKUP DATABASE successfully processed 17675 pages in 7.718 seconds (17.890 MB/sec).

- BACKUP LOG [AdventureWorks] TO
  - DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL10.SQLO8DE01\MSSQL\Backup\AdventureWorks4MirrorLog.bak'
- GO
  - -- BACKUP LOG successfully processed 1849 pages in 0.862 seconds (16.757 MB/sec).
**Restore to Mirror**

- RESTORE FILELISTONLY
  - FROM DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL10.SQL08DE01\MSSQL\Backup\AdventureWorks4Mirror.bak'

- RESTORE DATABASE AdventureWorks
  - FROM DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL10.SQL08DE01\MSSQL\Backup\AdventureWorks4Mirror.bak'
  - WITH NORECOVERY,
  - MOVE 'AdventureWorks_Data' TO 'C:\Program Files\Microsoft SQL Server\MSSQL10.SQL08DE02\MSSQL\DATA\AdventureWorks_Data.mdf',
  - MOVE 'AdventureWorks_Log' TO 'C:\Program Files\Microsoft SQL Server\MSSQL10.SQL08DE02\MSSQL\DATA\AdventureWorks_Log.ldf'
- GO

- RESTORE LOG AdventureWorks
  - FROM DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL10.SQL08DE01\MSSQL\Backup\AdventureWorks4MirrorLog.bak'
  - WITH FILE = 1, NORECOVERY;
- GO
  - -- Processed 0 pages for database 'AdventureWorks', file 'AdventureWorks_Data' on file 1.
  - -- RESTORE LOG successfully processed 1849 pages in 1.083 seconds (13.338 MB/sec).
## Database Mirroring History

### Server Instance:
- **REM1237433\SQL08DE01**

### Database:
- **AdventureWorks**

### Filter list by:
- **Last two hours**

### History:

<table>
<thead>
<tr>
<th>Time Recorded</th>
<th>Role</th>
<th>Mirrored</th>
<th>Witness</th>
<th>Unseen</th>
<th>Time</th>
<th>Send</th>
<th>New</th>
<th>Old</th>
<th>Unseen</th>
<th>Time</th>
<th>Rest</th>
<th>Mirror</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/7/2009 10:46:29 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:46:08 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:45:38 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:45:08 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:44:38 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:44:08 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:43:38 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:43:08 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:42:38 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:42:08 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
<tr>
<td>2/7/2009 10:41:38 PM</td>
<td>Prim.</td>
<td>Syn.</td>
<td>Con.</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0 K...</td>
<td>0:0:0...</td>
<td>0 KB</td>
<td>0:0:0...</td>
</tr>
</tbody>
</table>

### Set Warning Thresholds

Thresholds and alerts should be defined at both the principal and mirror server instances so that you continue to receive alerts after a failover.

#### Connected to REM1237433\SQL08DE01
- Connected

#### Connected to REM1237433\SQL08DE02
- Connected

<table>
<thead>
<tr>
<th>Thresholds</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Warning</strong></td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>Warn if the unfiltered log exceeds</td>
</tr>
<tr>
<td>Warn if the unfiltered log exceeds</td>
</tr>
<tr>
<td>Warn if the age of the oldest unfiltered log exceeds</td>
</tr>
<tr>
<td>Warn if the mirror commit overhead exceeds</td>
</tr>
</tbody>
</table>
Ensure that security is configured for mirroring the database.

Server network addresses:
- Principal: TCP://REM12374333.sql.autodesk.com:1440
- Mirror: TCP://REM12374333.sql.autodesk.com:1450
- Witness: TCP://REM12374333.sql.autodesk.com:1470

Note: Use fully-qualified TCP addresses. For example:
TCP://svs.autodesk.com:502

Operating mode:
- High performance (asynchronous) - Commit changes at the principal and then transfer them to the mirror.
- High safety (synchronous) - Always commit changes at both the principal and mirror.
- High safety with automatic failover (synchronous) - Requires a witness server instance. Commit changes at both the principal and mirror if both are available. The witness controls automatic failover to the mirror if the principal becomes unavailable.

Status: Synchronized; the databases are fully synchronized
Unhandled exception has occurred in your application. If you click Continue, the application will ignore this error and attempt to continue. If you click Quit, the application will close immediately.

A transport-level error has occurred when sending the request to the server. (provider: Shared Memory Provider, error: 0 - No process is on the other end of the pipe.)

### SQL2008 Connection Test Program

#### FillBy

<table>
<thead>
<tr>
<th>ProductID</th>
<th>Name</th>
<th>ProductNumber</th>
<th>SHOWDATETIME</th>
<th>SERVERNAME</th>
<th>SPID</th>
</tr>
</thead>
<tbody>
<tr>
<td>320</td>
<td>Chainring Bolts</td>
<td>CB-2903</td>
<td>Feb 8 2009 12:16:08:030AM</td>
<td>REM12374333\SQL08DE02</td>
<td>57</td>
</tr>
<tr>
<td>321</td>
<td>Chainring Nut</td>
<td>CN-6137</td>
<td>Feb 8 2009 12:16:08:030AM</td>
<td>REM12374333\SQL08DE02</td>
<td>57</td>
</tr>
<tr>
<td>322</td>
<td>Chainring</td>
<td>CR-7833</td>
<td>Feb 8 2009 12:16:08:030AM</td>
<td>REM12374333\SQL08DE02</td>
<td>57</td>
</tr>
<tr>
<td>323</td>
<td>Crown Race</td>
<td>CR-9981</td>
<td>Feb 8 2009 12:16:08:030AM</td>
<td>REM12374333\SQL08DE02</td>
<td>57</td>
</tr>
<tr>
<td>324</td>
<td>Chain Stays</td>
<td>CS-2812</td>
<td>Feb 8 2009 12:16:08:030AM</td>
<td>REM12374333\SQL08DE02</td>
<td>57</td>
</tr>
</tbody>
</table>

Press Retrieve to refresh the list.
DB Mirroring 2005 Benchmarks

Repl_rows stored procedure 9999999
DB Mirroring 2008 Benchmarks

Send Rate

41% More
DB Mirroring 2005 Benchmarks

Restore Rate

```
insert into #test exec sys.sp_0monmonitorresults @database_name='AdventureWorks',
    Result=9, Update_table = 0

select avg(recovery_rate) from #test
--Pacific Daylight Time - UTC - 7 hours
where datename(hh-mm, time_recordable) between '2008-09-29 14:21:00' and '2008-09-29 19:56:00'
```

<table>
<thead>
<tr>
<th>Olds</th>
<th>Unrec</th>
<th>T</th>
<th>Restore Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>463 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>507 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>478 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>471 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>128 KB</td>
<td>489 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>477 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>464 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>438 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>451 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>485 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>447 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>498 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>498 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>465 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>304 KB</td>
<td>536 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>476 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>473 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>492 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>492 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>0 KB</td>
<td>465 KB/sec</td>
<td></td>
</tr>
<tr>
<td>00:00</td>
<td>64 KB</td>
<td>468 KB/sec</td>
<td></td>
</tr>
</tbody>
</table>
DB Mirroring 2008 Benchmarks

Restore Rate

52% more
DB Mirroring 2005 Benchmarks
DB Mirroring Benchmark Summary

Overall **send rate** 41 % faster (2005 vs 2008)
Overall **restore rate** 52% faster (2005 vs 2008)

Overall availability topology inherits the restore rate ~ 50% more available

Example:
- 743 ms Transaction (in 2005 mirroring configuration)
- 379 ms Transaction (in 2008 mirroring configuration)

In failover, what would have been 20 seconds before mirror becomes principal
turns into roughly 8-10 seconds (or 3 seconds turns into 1.5 seconds, so on).
Snapshots
CREATE DB 'xyz' AS SNAPSHOT OF AdventureWorks

--- Reverting --- if needed ---
RESTORE DB AdventureWorks
FROM DATABASE_SNAPSHOT = 'xyz'

Transaction Users

Point-in-time Reporting Users (Read-Only)
Source Data Pages

Sparse File Pages

Empty Sparse file of a Snapshot just created (no updates to original data pages have occurred yet)
Source

SQL Server

AdventureWorks DB

Source Data Pages

Sparse File Pages

Copy of original pages for snapshot only when a page is changed (Copy-on-write)

System Catalog of changed pages
Source Data Pages

SELECT .....data....... FROM AdventureWorks SNAPSHOT

Snapshot Users

SQL Server 2008

AdventureWorks DB

System Catalog of changed pages

Sparse File Pages

Copyright 2009 all rights reserved
SQL Server 2008

Snapshot

AdventureWorks DB

Data Pages

System Catalog of changed pages

Copy of original pages for snapshot use only when a page is changed (Copy-on-write)

Committed

Update Row

Sparse File Pages

Transactional Users

translog

AdventureWorks DB
UPDATE AWSource.tableX
set xyz = ...
FROM AWSnapshot6:00AM.tableX

Restore from Any Point-in-time Snapshot if needed
SQL Server 2008

Source Server

AdventureWorks DB

Database Snapshot

SAFE GUARD Snapshot (Before the mass changes)

Generating Mass Changes

Users

UPDATE AWSource.tableX
set xyz = ...
FROM AWSafeguard6:00AM.tableX

Restore from snapshot (if changes are not correct)
SQL Server 2008

Source Server

AdventureWorks DB

Database Snapshot

BEFORE TEST Snapshot

Restore from the BEFORE snapshot after testing cycle is complete and can continue with next test.
SQL Server 2008

Source Server

AdventureWorks DB

translog

Transactional Users

Point-in-time Reporting Users (Read-Only)

6:00AM Reporting Snapshot
12:00PM Reporting Snapshot
6:00PM Reporting Snapshot
12:00AM Reporting Snapshot
If this server node becomes the mirror.
PH Topology With Snapshots

Instance: SQL2008xyz
Endpoint Name: “endpoint4mirroring”
Role: PARTNER

SQL Server 2008
Principal Server
Active
Clustered
Adventure Works DB
translog

Instance: SQL2008zzz
Endpoint Name: “endpoint4mirroring”
Role: PARTNER

SQL Server 2008
Mirror Server
Adventure Works DB
translog
Database Snapshot
Network
Reporting Users

OLTP Application
Replication

Reporting Users
DB Mirror and Replication

Publisher

Principal Server

Mirror Server

Subscriber

Distributor

Principal Server

Mirror Server

Witness Server

Subscriber

Copyright 2009 all rights reserved
<table>
<thead>
<tr>
<th>Data Access</th>
<th>Latency</th>
<th>Autonomy</th>
<th>Sites (locations)</th>
<th>Frequency</th>
<th>Network</th>
<th>Machines</th>
<th>Owner</th>
<th>Other</th>
<th>REPLICATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Only</td>
<td>short</td>
<td>high</td>
<td>many</td>
<td>high</td>
<td>fast/</td>
<td>1</td>
<td>1 OLTP</td>
<td></td>
<td>Central Publisher Transactional repl filter by region</td>
</tr>
<tr>
<td>Reporting</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>stable</td>
<td>server/site</td>
<td>site</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Only</td>
<td>long</td>
<td>high</td>
<td>many</td>
<td>low</td>
<td>fast/</td>
<td>1</td>
<td>1 OLTP</td>
<td></td>
<td>Central Publisher Snapshot repl filter by region</td>
</tr>
<tr>
<td>Reporting</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>stable</td>
<td>server/site</td>
<td>site</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Mostly</td>
<td>short</td>
<td>high</td>
<td>&lt; 10</td>
<td>medium</td>
<td>fast/</td>
<td>1</td>
<td>1 OLTP</td>
<td></td>
<td>Regional updates on one table</td>
</tr>
<tr>
<td>A few updates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>stable</td>
<td>server/site</td>
<td>site</td>
<td></td>
<td>Central Publisher Transactional repl Updating Subs</td>
</tr>
<tr>
<td>Read Mostly</td>
<td>medium</td>
<td>high</td>
<td>&lt; 10</td>
<td>medium</td>
<td>slow/</td>
<td>1</td>
<td>All</td>
<td></td>
<td>Regional update all tables</td>
</tr>
<tr>
<td>A few updates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>unrelia</td>
<td>server/site</td>
<td>update</td>
<td></td>
<td>Central Publisher Merge repl</td>
</tr>
<tr>
<td>Read equal</td>
<td>short</td>
<td>high</td>
<td>&lt; 10</td>
<td>medium</td>
<td>fast/</td>
<td>1</td>
<td>All</td>
<td></td>
<td>Regional update all tables</td>
</tr>
<tr>
<td>Equal updates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>stable</td>
<td>server/site</td>
<td>update</td>
<td></td>
<td>Peer-to-Peer Transactional repl</td>
</tr>
<tr>
<td>Inserts (new</td>
<td>short</td>
<td>high</td>
<td>many</td>
<td>high</td>
<td>fast/</td>
<td>1</td>
<td>1 report</td>
<td></td>
<td>Central Subscriber Transactional repl</td>
</tr>
<tr>
<td>orders)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>stable</td>
<td>server/site</td>
<td>site</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hot/Warm</td>
<td>Very</td>
<td>high</td>
<td>&lt; 2</td>
<td>high</td>
<td>fast/</td>
<td>1</td>
<td>1 OLTP</td>
<td>Fail-over</td>
<td>Central Publisher Remote Distributor Transactional repl</td>
</tr>
<tr>
<td>Spare</td>
<td>short</td>
<td></td>
<td></td>
<td></td>
<td>stable</td>
<td>server/site</td>
<td>site</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Questions

Is there any time left????

Send your questions to:
pbertucci@dbarchitechs.com
or
tgerardin@dbarchitechs.com
SQL SHOT!

Database Products

MS SQL Server, Sybase and Oracle

<table>
<thead>
<tr>
<th>Spid</th>
<th>Login</th>
<th>Host</th>
<th>Application</th>
<th>Start</th>
<th>SQL Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>52</td>
<td>Thierry</td>
<td>SSV_DEMO</td>
<td>SQL-32</td>
<td>2009-08-19 08:24:51:017</td>
<td>waitfor delay @delay</td>
</tr>
<tr>
<td>53</td>
<td>Greg</td>
<td>SSV_DEMO</td>
<td>SQL-32</td>
<td>2009-08-19 08:24:53:313</td>
<td>update Dbarc_cnt set id = id + 1 where code = @code</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Spid</th>
<th>Login</th>
<th>Host</th>
<th>Application</th>
<th>Start</th>
<th>SQL Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>Paul</td>
<td>SSV_DEMO</td>
<td>SQL-32</td>
<td>2009-08-19 08:24:47:017</td>
<td>update code</td>
</tr>
<tr>
<td>54</td>
<td>Yves</td>
<td>SSV_DEMO</td>
<td>SQL-32</td>
<td>2009-08-19 08:24:46:017</td>
<td>update code</td>
</tr>
<tr>
<td>53</td>
<td>Greg</td>
<td>SSV_DEMO</td>
<td>SQL-32</td>
<td>2009-08-19 08:24:30:467</td>
<td>waitfor</td>
</tr>
<tr>
<td>52</td>
<td>Thierry</td>
<td>SSV_DEMO</td>
<td>SQL-32</td>
<td>2009-08-19 08:24:31:160</td>
<td>update code</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elapsed Time</th>
<th>Started</th>
<th>Completed</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>08:20</td>
<td>08:21</td>
<td>100</td>
</tr>
<tr>
<td>50</td>
<td>08:22</td>
<td>08:23</td>
<td>50</td>
</tr>
</tbody>
</table>

Legend
- Ready