



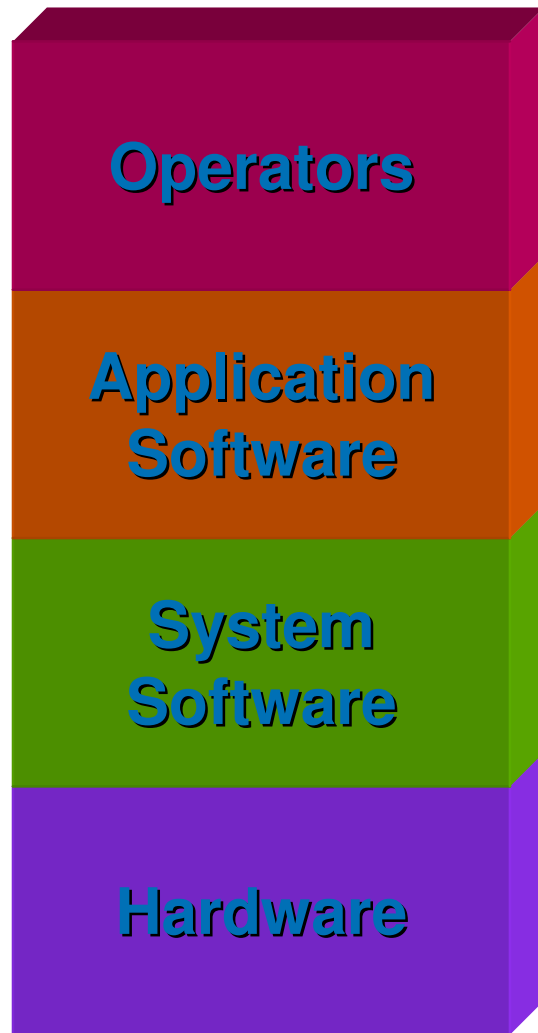
High Availability Technologies In SQL Server 2000 & SQL Server 2005

Prem Mehra

Program Manager

SQL Server Storage Engine , Microsoft Corporation

Availability “Stack”



- Majority of downtime is ascribed to operators
- Application Software can have dramatic impact on fault tolerance
- System Software can help at all levels
 - More later
- Hardware is very reliable at this point

Barriers To Availability

Many barriers

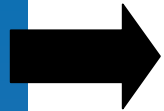
Only some are addressable by DBMS technology

Be sure to consider people, planning, and procedures

- Microsoft SQL Server 2005 gives you greatly improved tools to overcome these barriers
 - Database Server Failure or Disaster
 - User or Application Error
 - Data Access Concurrency Limitations
 - Database Maintenance and Operations
 - Upgrades
 - Availability at Scale
 - Tuning

Barriers To Availability

As addressed in SQL Server 2005



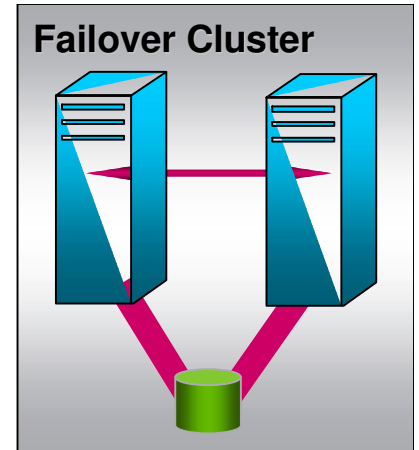
Database Server Failure or Disaster

- Failover Clustering
- Database Mirroring
 - Transparent Client Redirect
- User or Application Error
- Data Access Concurrency Limitations
- Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning

Failover Clustering

Microsoft Cluster Services

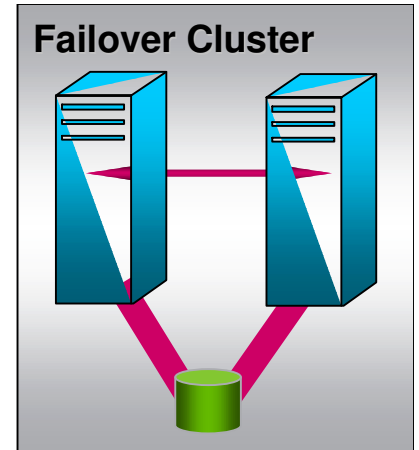
- Hot Standby – Automatic failover
- Built on Microsoft Cluster Services (MSCS)
 - Multiple nodes provide availability, transparent to client
 - Automatic detection and failover
 - Requires certified hardware
 - Supports many scenarios: Active/Active, N+1, N+1
- Zero work loss, zero impact on throughput
- Instance Failover – entire instance works as a unit
- Single copy of instance databases
- Available since SQL Server 7.0
- Standby is not available for reporting, queries, etc.
 - May support other instances



Failover Clustering

SQL Server 2005

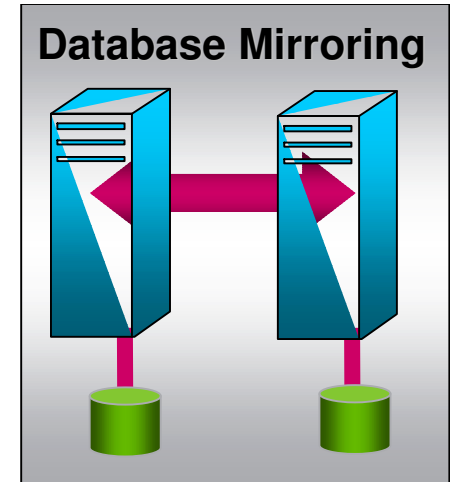
- Further refined in SQL Server 2005
- More nodes
 - Match operating system limits
- Unattended setup
- Support for mounted volumes (Mount Points)
- All SQL Server services participate
 - Database Engine, SQL Server Agent, Analysis Services, Full-Text Search, etc.



Database Mirroring

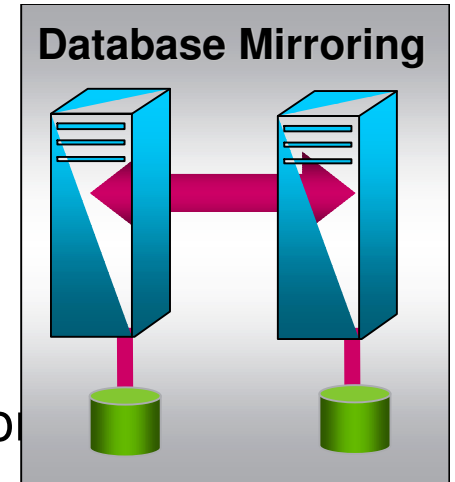
New for SQL Server 2005

- Instant Standby
- Conceptually a fault-tolerant server
 - Building block for complex topologies
- Database Failover
 - Very Fast ... less than three seconds
 - Zero data loss
- Automatic or manual failover
 - Automatic re-sync after failover
- Automatic, transparent client redirect



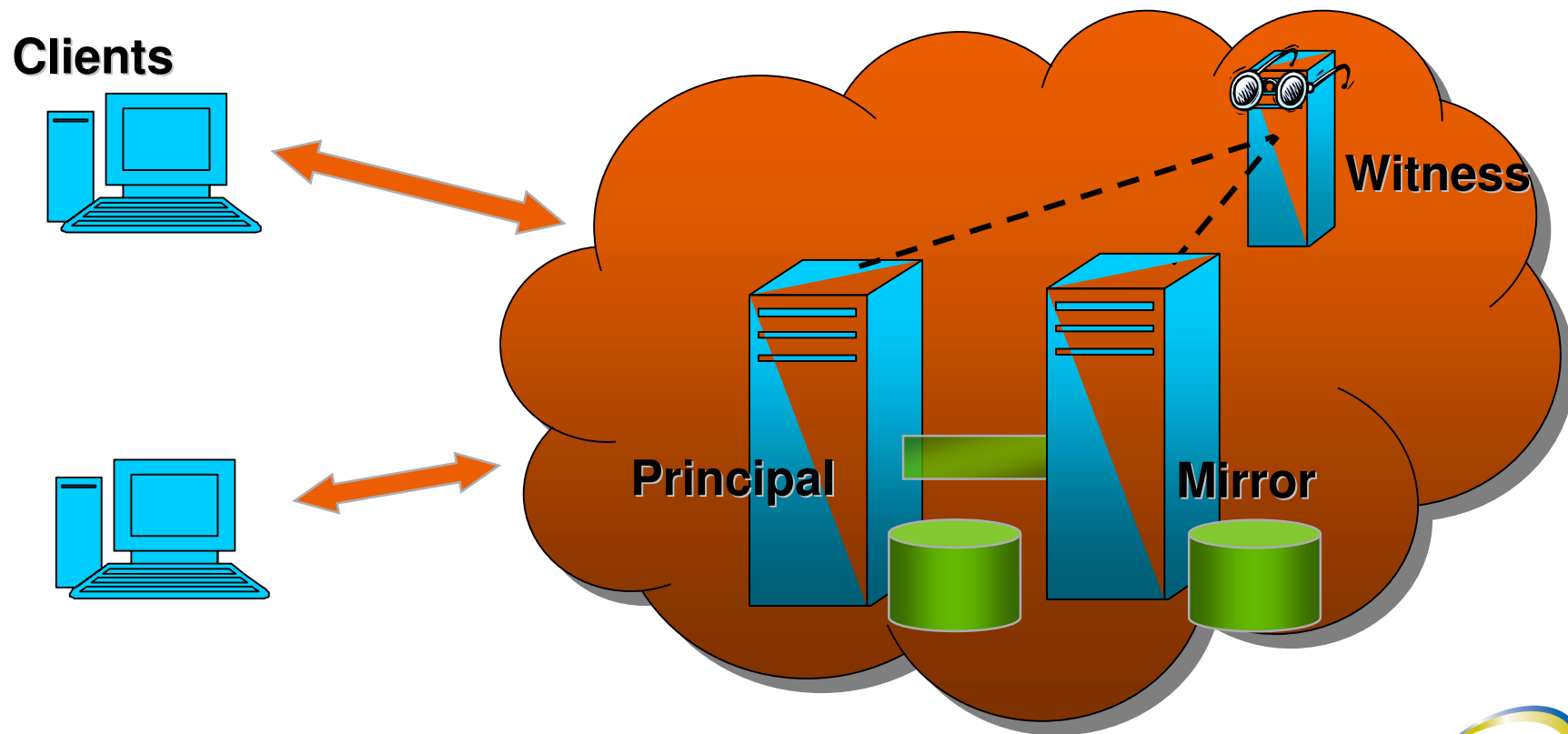
Database Mirroring

- Hardware
 - Works with standard computers, storage, and network
 - No shared storage components, virtually no distance limitations
- Impact to transaction throughput
 - Zero to minimal, depending on environment / workload



Database Mirroring

- Fault Tolerant Virtual Database



Initiating a Database Mirroring Session

- On the principal server, back up the database

```
BACKUP DATABASE AdventureWorks  
TO DISK = 'c:\AdventureWorks.bak'
```
- On the future mirror server, restore the database

```
RESTORE DATABASE AdventureWorks  
FROM DISK = 'z:\AdventureWorks.bak'  
WITH NORECOVERY
```
- On the mirror, set the principal server as a failover partner

```
ALTER DATABASE AdventureWorks  
SET PARTNER = 'sqlDbEngine05\MSSQLSERVER'
```
- On the principal server set the mirror server as the second failover partner

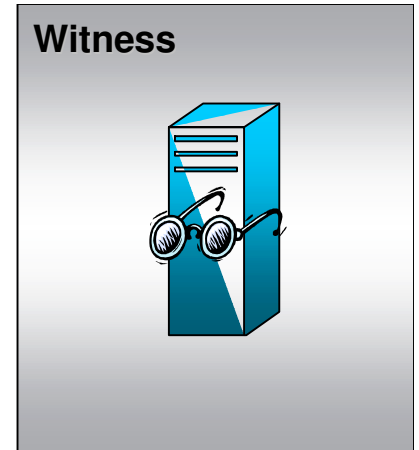
```
ALTER DATABASE AdventureWorks  
SET PARTNER = 'sqlDbEngine06\MSSQLSERVER'
```
- To fail over: On the principal

```
ALTER DATABASE db  
SET PARTNER FAILOVER
```
- To view metadata

```
SELECT * FROM SYS.DATABASE_MIRRORING
```

Witness and Quorum

- Sole purpose of the Witness is to provide *automatic* failover
- To survive the loss of *one* server you must have at least *three*
- Prevents “split brain”
 - Does a lost connection mean the partner is down or is the network down?
- To become the Principal, a server must talk to at least one other server



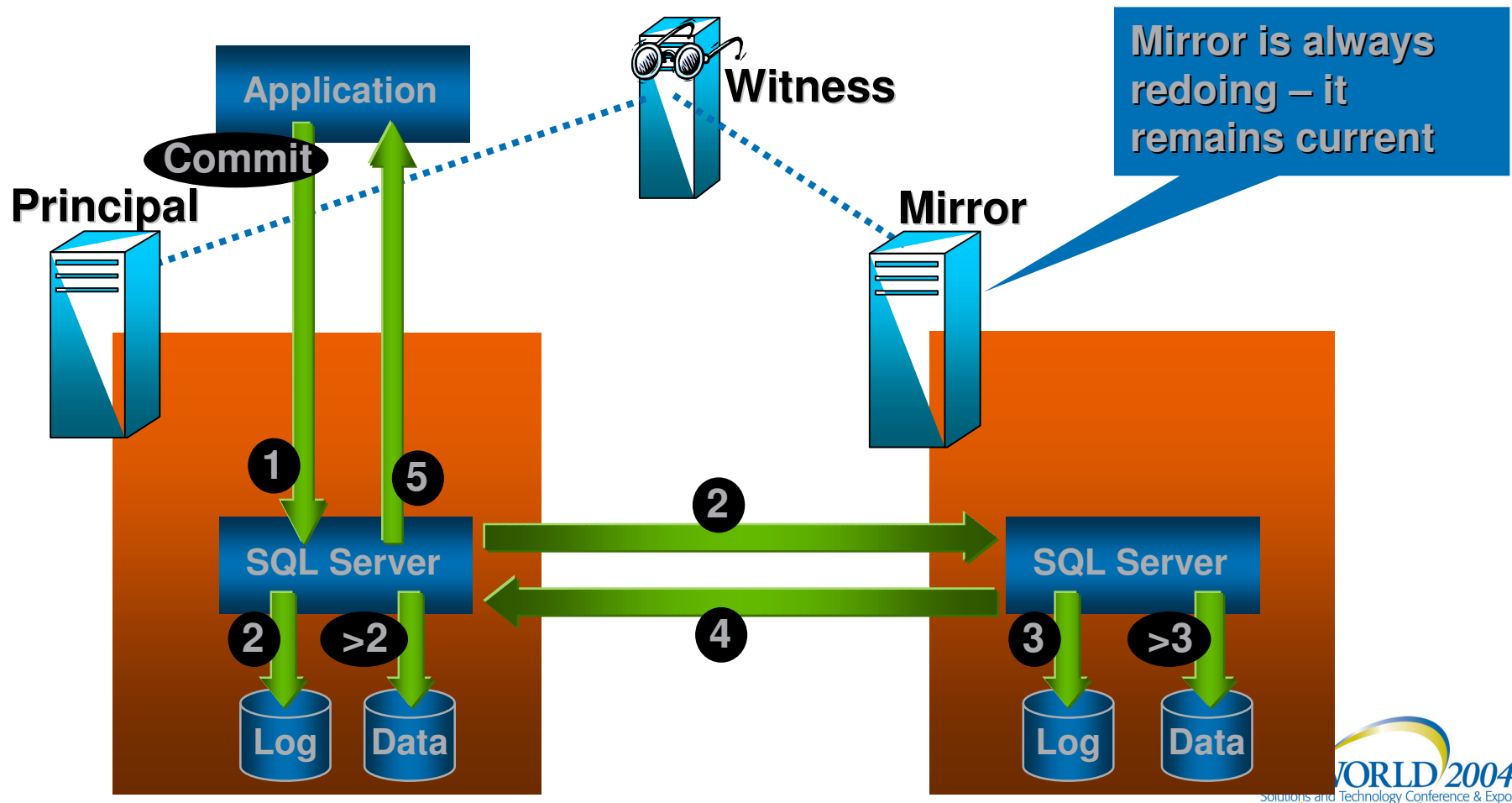
Witness

- Witness is an instance of SQL Server 2005
- Single witness for multiple sessions
- Consumes very little resources
- Not a single point of failure
 - Partners can form quorum on their own



Database Mirroring

How it works



Safety / Performance

- There is a trade-off between performance and safety
- Database Mirroring has two safety levels
 - FULL – commit when logged on Mirror
 - Allows automatic failover
 - No data loss
 - OFF – commit when logged on Principal
 - System does its best to keep up
 - Prevents failover; to make mirror available
 - Must 'force' service
 - Or terminate Database Mirroring session

Transparent Client Redirect

- No changes to application code
- Client automatically redirected if session is dropped
 - Client library is aware of Principal and Mirror servers
 - Upon initial connect to Principal, library caches Mirror name
 - When client attempts to reconnect
 - If Principal is available, connects
 - If not, client library automatically redirects connection to Mirror

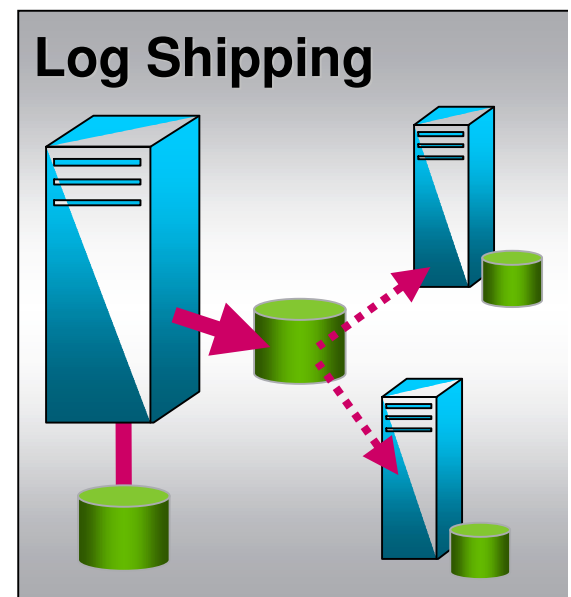
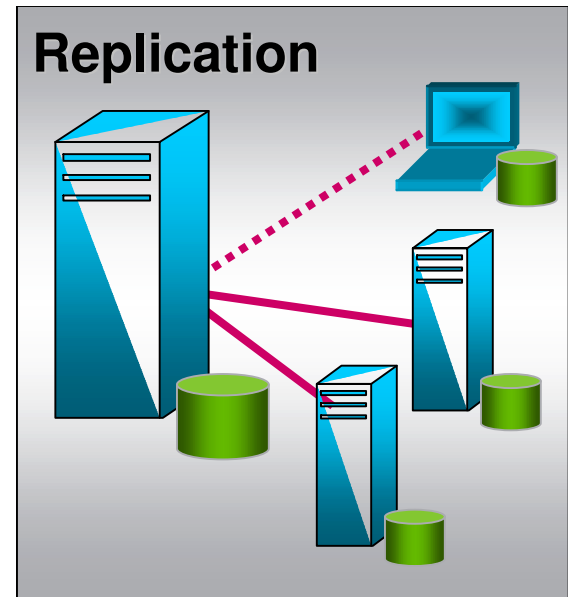
Failover Solutions At A Glance

- Both Provide
 - Automatic detection and failover
 - Manual failover
 - Transparent client connect
 - Zero work loss
 - Database Views mitigate DBA or application error
-
- | | |
|--|--|
| <ul style="list-style-type: none">• Failover Clustering<ul style="list-style-type: none">– System scope– Certified hardware– Fast failover– No reporting on standby– Single copy of database | <ul style="list-style-type: none">• Database Mirroring<ul style="list-style-type: none">– Database scope– Standard servers– Fastest failover– Limited reporting on standby– Duplicate copy of database |
|--|--|

Warm Standby Solutions

Replication and Log Shipping

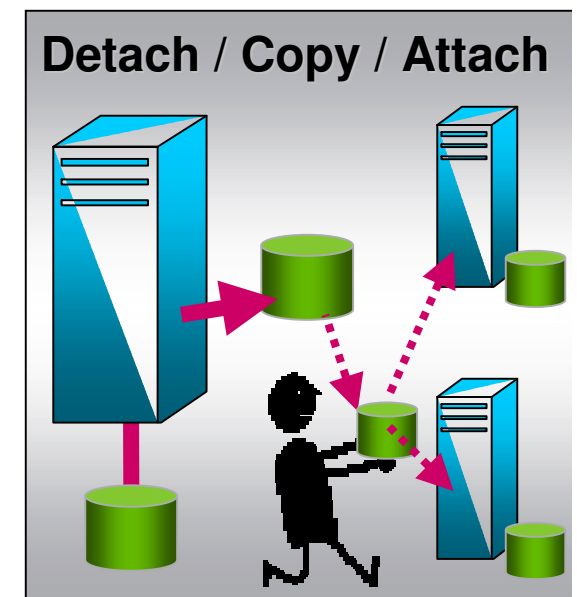
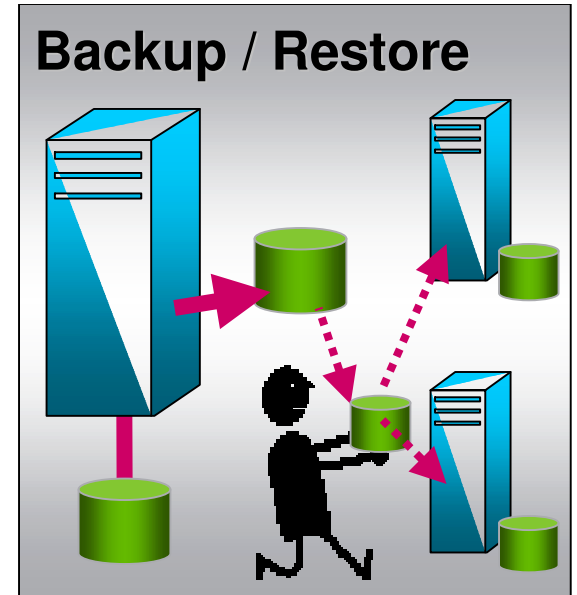
- Both Provide Multiple copies and Manual failover
- Replication – since SQL Server 6.0
 - Primarily used where availability is required in conjunction with scale out of read activity
 - Failover possible; a custom solution
 - Not limited to entire database; Can define subset of source database or tables
 - Copy of database is continuously accessible for read activity
 - Latency between source and copy can be as low as seconds
- Log Shipping
 - Basic idea: Backup, Copy, Restore Log will always be supported
 - But no more investment in the scripts
 - Database scope
 - Database accessible but read-only
 - Users must exit for next log to be applied



Cold Standby Solutions

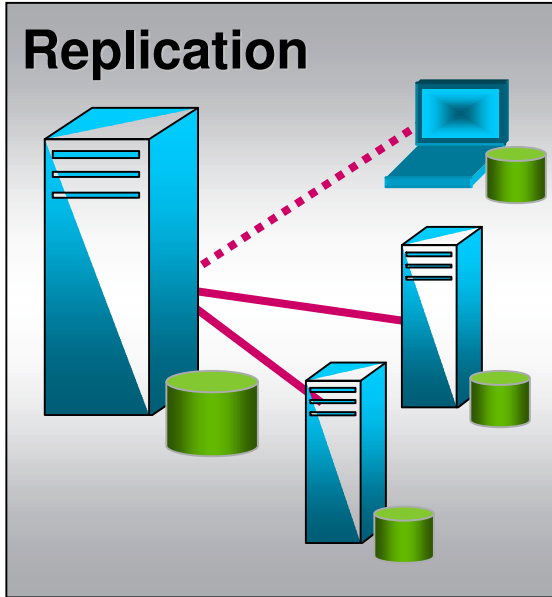
Backup / Restore and Detach / Copy / Attach

- Both Provide
 - Manual detection and failover
 - Potential for some work loss
 - Whole-database scope
 - Standard servers
 - Limited reporting on standby
 - Duplicate copy of database
 - Client must know where to re-connect
 - Slowest failover – Most downtime
- Backup / Restore
 - Smaller size – only used pages are copied
 - Log backups allow restore to point in time
 - Longer restore time
- Detach / Copy / Attach
 - Copies entire files
 - No possibility of rolling forward subsequent logs



Complementary Technologies

Replication



- Maximize availability for
 - Scale out
 - Offload primary data platform
 - Heavy reporting
 - Mobile/disconnected users
 - Autonomous business units that share data

Failover Solutions



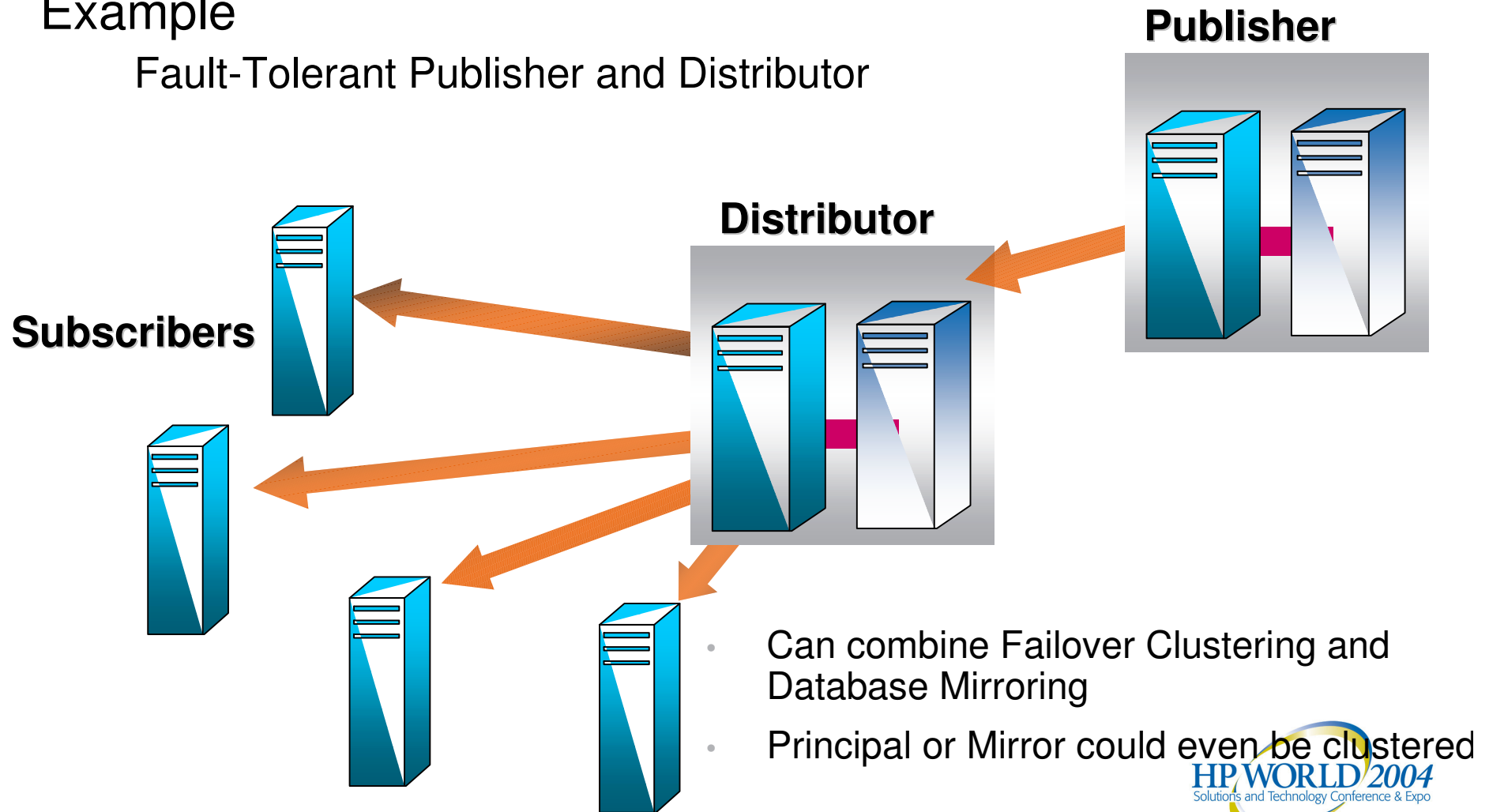
- Maximize availability of critical systems
 - Designed for failover
 - Fast, automatic
 - Zero data loss
 - Transactionally current
 - Masks planned and unplanned downtime

Complementary Technologies

Failover Solution + Replication

Example

Fault-Tolerant Publisher and Distributor



Barriers To Availability

As addressed in SQL Server 2005

- Database Server Failure or Disaster
- User or Application Error
 - Database Snapshot
- Data Access Concurrency Limitations
- Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning

User Error

- Users, applications, and DBAs do make errors
 - Good management tools, automation, and defined procedures help prevent many potential mistakes
 - User errors are the largest cause of lack of availability
- But what do you do when someone performs a risky operation and discovers too late that it needs to be rolled back?
- Restore from backup ..
 - You do have a recent backup, right?
 - And you do verify your backups are good, right?
 - And you do have time to do the Restore?

Database Snapshot

New in SQL Server 2005

- Database Snapshots allow recovery from user errors by allowing the database to go back in time
- Works with
 - Single server
 - Database Mirroring
 - Failover Cluster
- Does not work with Log Shipping secondary

Database Snapshot

- Snapshot of an entire database at a point in time
 - Created instantly
 - Read only
 - Snapshot must be created before the error
- Base database continues to change
 - Database Snapshot does not restrict the base database
- Multiple Snapshots are allowed
- Database Snapshots can exist forever
 - Constrained by resources

Database Snapshot

Uses

- Recover from User, Application, or DBA error
 - Revert database to previously created Database Snapshot
 - Takes the database back in time
 - Very fast, no restoring of backups required
- Static, time-consistent copy for reports
- With Database Mirroring enables reporting on the standby
 - No increase in failover time

Database Snapshot

Technology

- Extremely space efficient
- Does not require a complete copy of the data
 - Shares unchanged pages of the database
 - Requires extra storage only for changed pages
- Uses a “copy-on-write” mechanism
- Database Snapshot may affect performance on the base database

Database Snapshot

Syntax Examples

- To create a database snapshot

```
CREATE DATABASE mydbSnap0600  
ON (<filelist>)  
AS SNAPSHOT OF mydb
```

- To drop a database snapshot

```
DROP DATABASE mydbSnap0600
```

- To revert a database to a snapshot

```
RESTORE DATABASE mydb  
FROM DATABASE_SNAPSHOT  
= 'mydbsnap0600'
```

Database Snapshot

How it works

```
CREATE DATABASE dbSnap AS SNAPSHOT OF mydb
```

```
USE mydb
```

```
UPDATE (pages 4, 9, 10)
```

mydb – Database

Page

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----

```
USE mydbSnap
```

```
SELECT (pages 4, 6, 9, 10, 14)
```

dbSnap – Read-Only Database Snapshot

Database Snapshot

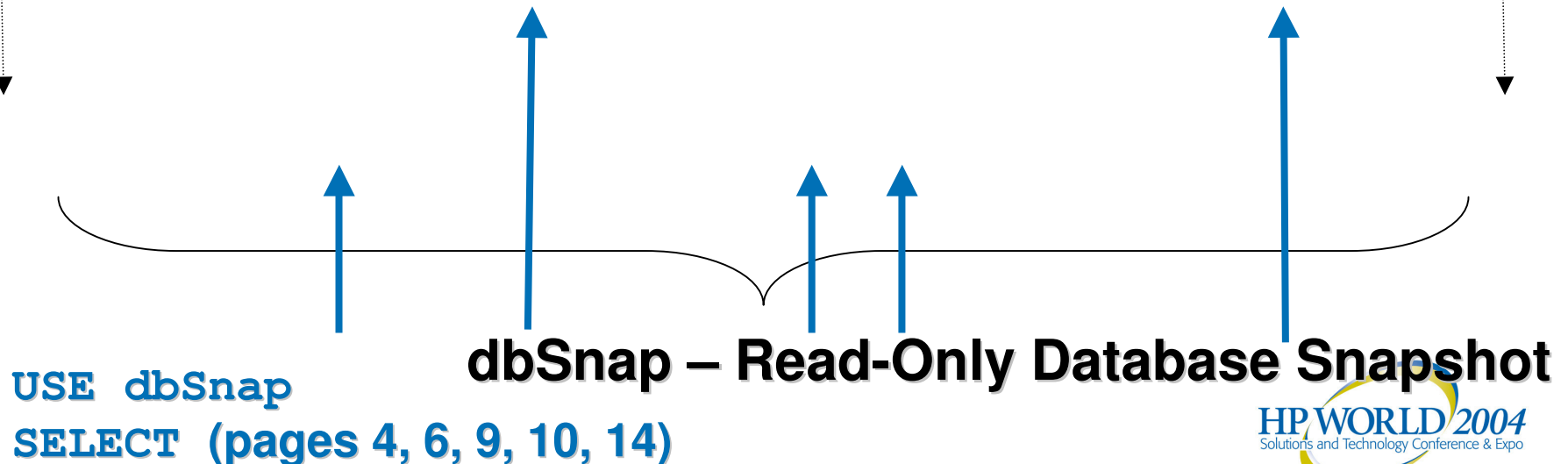
How it really works

```
CREATE DATABASE dbSnap AS SNAPSHOT OF mydb
USE mydb
UPDATE (pages 4, 9, 10)
```

mydb – Database

Page

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----

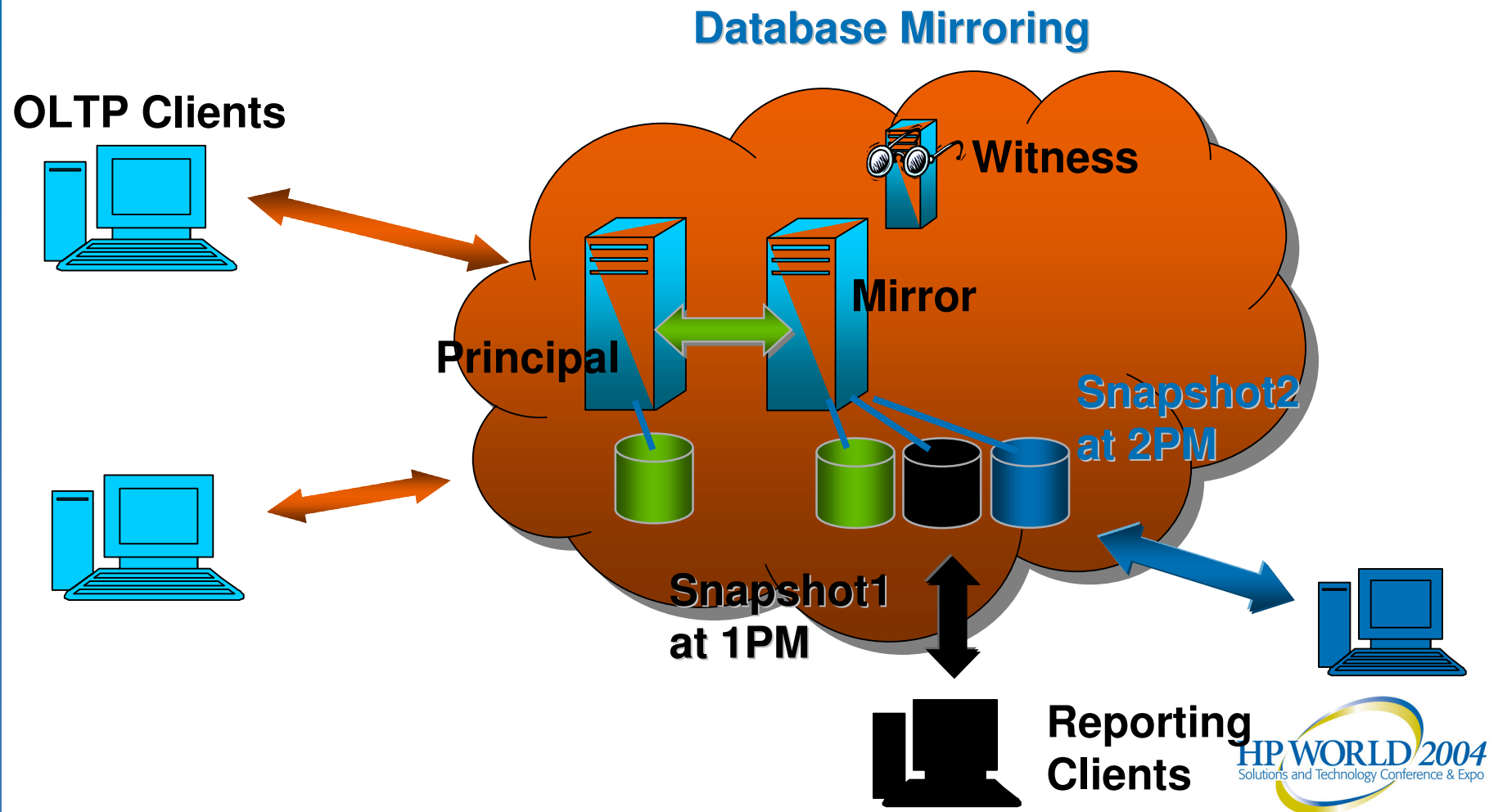


USE dbSnap

SELECT (pages 4, 6, 9, 10, 14)

Reporting on a Mirror

- Use Database Snapshots on Mirror



Barriers To Availability

As addressed in SQL Server 2005

- Database Server Failure or Disaster
- Application or User Error
- ➔ Data Access Concurrency Limitations
 - Snapshot Isolation
 - Online Index Operations
- Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning

Transaction Isolation Levels

- Isolation levels control interaction of many users working simultaneously with transactional data
- Trade-off between concurrency and correctness
- SQL-92 specifies four isolation levels
 - Serializable
 - Least concurrency, most restrictive, always 'correct'
 - Repeatable Read
 - Read Committed
 - Read Uncommitted
 - Most concurrency, least restrictive, more anomalies are possible
- SQL Server has implemented these all along
 - Uses 'pessimistic locking', locking the row, page, or table assuming another user will try to access it

Snapshot Isolation

New in SQL Server 2005

- Two new 'snapshot isolation' levels
- Increased data availability for read applications
 - Allows non-blocking consistent reads in an online transaction processing (OLTP) environment
 - Writers do not block readers; readers do not block writers
 - Consistency of aggregates without using higher isolation levels
 - AVG, COUNT, SUM, etc.

Snapshot Isolation

New in SQL Server 2005

- Increases concurrency and data availability while reducing deadlocks
 - Non-blocking consistent reporting and ad-hoc queries
- Uses row-level versions on update and delete to keep copies of the versions of the row other users might want to see
 - Doesn't lock the row
- Permits writes, which can cause conflicts
 - BUT...includes mandatory conflict detection

Snapshot Isolation

Transaction-Level 'Snapshot Isolation'

- 'Snapshot Isolation' – Transaction-level

```
SET TRANSACTION ISOLATION LEVEL  
SNAPSHOT
```

- Uses row-level versioning
- Read operations do not acquire locks
- When referencing a row modified by another transaction will retrieve the committed version of the row that existed when the snapshot transaction started

Snapshot Isolation

Statement-Level 'Snapshot Isolation'

- Read Committed Snapshot – Statement-level
`SET TRANSACTION ISOLATION LEVEL READ COMMITTED`
- While `READ_COMMITTED_SNAPSHOT` database option is set to ON, automatically get non-locking `READ COMMITTED`
 - Can greatly reduce locking / deadlocking without changing applications
 - Override with `READCOMMITTEDLOCK` hint
 - The query scan will run under the original flavor of locking-based read committed isolation
- When referencing a row modified by another transaction will retrieve the committed version of the row that existed when the statement started
- Writers do block writers

Snapshot Isolation

Example

```
CREATE TABLE t1 (c1 int unique, c2 int)
INSERT INTO t1 VALUES (1, 5)
```

Transaction 1

```
BEGIN TRAN
UPDATE t1
  SET c2 = 9
  WHERE c1 = 1

COMMIT TRAN
```

Transaction 2 (Snapshot Isolation)

```
SET TRANSACTION ISOLATION LEVEL
  SNAPSHOT

BEGIN TRAN

SELECT c2 FROM t1
  WHERE c1 = 1
  -- SQL Server returns 5

SELECT c2 FROM t1
  WHERE c1 = 1
  -- SQL Server returns 5

COMMIT TRAN

SELECT c2 FROM t1
  WHERE c1 = 1
  -- SQL Server returns 9
```

Transaction 3 (RCSI)

```
BEGIN TRAN

SELECT c2 FROM t1
  WHERE c1 = 1
  -- SQL Server returns 5

SELECT c2 FROM t1
  WHERE c1 = 1
  -- SQL Server returns 9

COMMIT TRAN

SELECT c2 FROM t1
  WHERE c1 = 1
  -- SQL Server returns 9
```

Time

Snapshot Isolation

Database-Level Settings

- Snapshot Isolation must be enabled at the database level



```
ALTER DATABASE mydatabase  
SET ALLOW_SNAPSHOT_ISOLATION ON
```

- To default all Read Committed operations to use Read Committed Snapshot

```
ALTER DATABASE mydatabase  
SET READ_COMMITTED_SNAPSHOT ON
```

- All row versions are stored in *tempdb*

SQL Server 2005 Isolation Levels

Isolation Levels	Possible Anomalies				Concurrency Control
	Dirty Read	Non-Repeatable Read	Phantoms	Update Conflict	
Read Uncommitted	Yes	Yes	Yes	No	
Read Committed 1 Locking 2 Snapshot 	No No	Yes Yes	Yes Yes	No No	Pessimistic Optimistic
Repeatable Read	No	No	Yes	No	Pessimistic
Snapshot 	No	No	No	Yes	Optimistic
Serializable	No	No	No	No	Pessimistic

**Detected and rolled back
No “Lost Update”!**

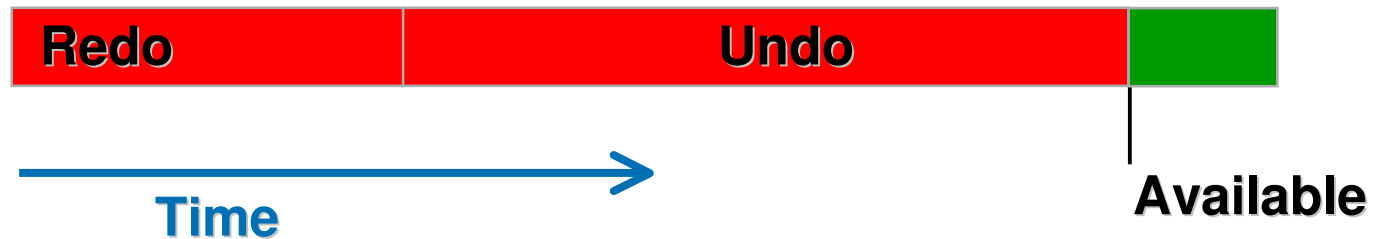
Online Index Operations

- Online Index Operations allow concurrent modification of the underlying table or index
 - Updates, Inserts, Deletes
- Online Index Maintenance
 - Create, Rebuild, Drop
 - Index-based constraints (PrimaryKey, Unique)
- Data definition language (DDL) is simple
- Online/Offline are both supported
- Updates incur some additional cost during an online index operation
 - Maintains old and new indexes

Fast Recovery

Restart or Restore

- SQL Server 2000
 - Database is available after Undo completes



- SQL Server 2005
 - Database is available when Undo begins



Barriers To Availability

As addressed in SQL Server 2005

- Database Server Failure or Disaster
- Application or User Error
- Data Access Concurrency Limitations
- • Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning

Database Maintenance and Operations

- Partial Availability
 - Database is available if primary filegroup is available
- Online Restore
 - Restore while database remains available
 - Works with all recovery models
- Backup and Restore
 - Data backups don't block log backups
 - Full-Text Catalog is backed up and restored as part of the database

Media Reliability Features

- Backup Media Mirroring
 - Can write backups to 4 destinations
- Enhanced Verification of Backups
 - RESTORE VERIFYONLY now checks everything it can, short of writing the data
- Database Page Checksums and Backup Checksums
 - Detects disk I/O errors not reported by the hardware or operating system
- Continue Past Errors Encountered by Restore
 - Allows the restore sequence to continue as far as possible...
 - ...then repair the database

More Operational Improvements...

- **Dedicated Administration Connection**
 - Provides DBA access to server regardless of load
 - No server restart to kill a runaway session
- **More configuration is dynamic**
 - No server restart for CPU affinity, AWE
 - Address Windowing Extensions (AWE)
 - Changes to physical size don't require downtime
 - Dynamically configurable (Min / Max)
 - Dynamically adjusts to "hot-add" memory
 - Requires Windows Server 2003
- **Instant file initialization**
 - With appropriate security, can bypass zeroing
 - For create database, add file, file grow, restore

Barriers To Availability

As addressed in SQL Server 2005

- Database Server Failure or Disaster
- Application or User Error
- Data Access Concurrency Limitations
- Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning

Upgrade Enhancements

- Software Upgrade
 - Re-architected – greatly reduces down time
 - Side-by-side installation
 - Resource database pre-built
 - Phased
 - Engine and Databases (< 3 minutes)
 - Other components complete upgrade online after databases are available (Replication, Workbench, etc.)
 - Database Mirroring allows “rolling upgrade”
 - Reduces downtime to seconds for service packs
 - Minimizes downtime for version upgrade
- Hardware Upgrade
 - Hot-add memory supported without server restart
 - Database Mirroring minimizes downtimes for other hardware upgrades, excluding disk

Barriers To Availability

As addressed in SQL Server 2005

- Database Server Failure or Disaster
- Application or User Error
- Data Access Concurrency Limitations
- Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning

Data Partitioning

- Partitioning breaks a single object into multiple manageable pieces
 - Transparent to the application
 - Allows easy management of very large tables and indexes
 - The row is the unit of partitioning
 - All partitions run on a single SQL Server database
- Partitions can be created or dropped with virtually no loss of availability to the table
 - Table fully available while loading, indexing a new partition
 - Create new and drop old partition quickly
 - Fastest possible load rates
- If all indexes are “aligned”, moving partitions in or out of a table is possible
 - “Sliding window” scenarios

Barriers To Availability

As addressed in SQL Server 2005

- Database Server Failure or Disaster
- Application or User Error
- Data Access Concurrency Limitations
- Database Maintenance and Operations
- Upgrades
- Availability at Scale
- Tuning



Database Tuning Advisor

- Formerly “Index Tuning Wizard”
- Bigger mission - not just indexes & indexed views
 - Supports partition and multi-database tuning
 - Exploits new indexing features
 - Index with included columns, ...
 - High availability recommendation mode recommends only indexes that can be built online
- Tune production servers using test servers
 - Exploits metadata-only database copy functionality
- Time bound tuning
 - Recommendation within specified wall-clock time
- May recommend table and index partitioning
- “What-if” analysis with DBA’s indexing suggestions

SQL Server 2005 Delivers Availability

- Makes High Availability accessible
- More choices
 - Hardware requirements
 - Database administration
 - Application development
- Richer Solutions → Manageable → Lower Cost
 - Failover
 - Distributed environments
 - Online management
 - Better concurrency
 - Partial availability
 - VLDB

Highly Available Databases

SQL Server 2005 Technology

- Database Server Failure or Disaster
 - Failover Clustering
 - Database Mirroring
- User or Application Error
 - Database Snapshot
- Data Access Concurrency Limitations
 - Snapshot Isolation
 - Online Index Operations
- Upgrade
 - Software and Hardware
- Tuning
 - Database Tuning Advisor
- Database Maintenance and Operations
 - Fast Recovery
 - Partial Availability
 - Online Restore
 - Media Reliability
 - Dedicated Administration Connection
 - Dynamic Configuration
- Availability at Scale
 - Data Partitioning

Community Resources

SQL Server Community sites

<http://www.microsoft.com/sql/community/default.mspx>

List of newsgroups

<http://www.microsoft.com/sql/community/newsgroups/default.mspx>

Locate Local User Groups

<http://www.microsoft.com/communities/usergroups/default.mspx>

Attend a free chat or Web cast

<http://www.microsoft.com/communities/chats/default.mspx>

<http://www.microsoft.com/usa/webcasts/default.asp>

HP WORLD 2004

Solutions and Technology Conference & Expo

Co-produced by:

