# 1 Introduction

Database Mirroring is one of the SQL Server 2005 key features in the area of high availability. It provides failover functionality at database level as opposed to clustering, which provides failover at instance level. Although it relies on transaction logs to provide mirroring functionality, it differs from SQL Server log shipping in the way that no transaction log backup is required.

In the event of a primary database server failure, the database mirror enables an application to reconnect quickly by reconnecting to a hot standby database server. Changes are reflected in real-time on the standby server and failover support can be set up to be performed automatically or manually.

Database Mirroring works with all standard hardware that supports SQL Server and no shared storage is required between the primary database server and the hot standby database server.

# 2 How It Works

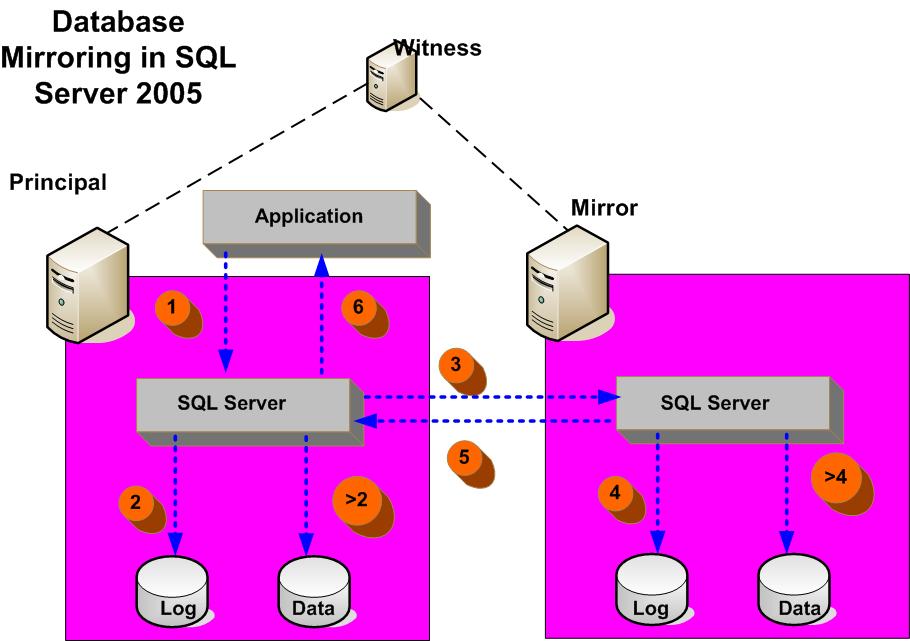


Figure 1: Database Mirroring Concept

The diagram above illustrates the concept of Database Mirroring in SQL Server 2005. Database Mirroring is made of two or three SQL Server instances depending on failover mode. Database Mirroring can be configured either in automatic or manual failover mode. A ‘Principal’ and a ‘Mirror’ SQL Server instances are part of any Database Mirroring configuration. For automatic failover mode a third SQL Server instance, a ‘Witness’, is essential. A witness is not required if Database Mirroring is set to have manual failover.

The principal instance is the instance that initially provides the database services. All application connections are made to the principal instance. The mirror instance maintains a copy of the principal database. The witness instance monitors the status of the principal and mirror instances. The witness doesn't really trigger the failover; all it does is to answer the question: “who do you see?” and “are you there?”

The principal and mirror instances are the ones that actually do the failover and decision making. The principal and mirror instances are each considered a partner in the database mirroring session. A database mirroring session consists of a relationship between the partner instances when they mirror a database from one partner to another.

In automatic failover mode each instance will get a vote to decide who will be a ‘Principal’. It takes two votes to decide on the principal instance. If communication between the principal and mirror is cut off, each instance will elect itself to be ‘Principal’. In this case the witness would cast the deciding vote.

A given partner instance may have the principal role for one database and a mirror role for a different database. A witness instance can handle multiple different pairings of partners.

Database Mirroring works by sending transaction logs between the principal and the mirror instances. When an application writes a request to a principal instance, that request is written to the principal database log file before it is written to the data file of the database. Next, that transaction is sent to the mirror database log file before it is written to the data file of the mirror instance. The ‘Mirror’, sends acknowledgement to the ‘Principal’ that the record has been received. In the case of commit operations, the ‘Principal’ will wait to receive the acknowledgement from the ‘Mirror’ before notifying the application that the operation is complete.

The principal database must be configured to be in the FULL recovery model, otherwise there will be no transaction logs to create the mirror. The mirror database must be initialized from a restore of the principal database with NORECOVERY, followed by restores in sequence of principal transaction log backups. Because the mirror database is in a recovering state, it cannot be accessed directly.

# Database Mirroring Operating Modes

Database Mirroring can be set to operate in one of the following operating modes:

* Synchronous with a witness
* Synchronous without a witness
* Asynchronous

Synchronous mode provides the most data protection. Failover can be performed either automatically or manually. This mode is achieved when the mirror database is in sync with the principal database (synchronized state).

Synchronous with a witness is a high availability operating mode. In this mode the failover is performed automatically when a principal instance fails. The mirror instance switches to the role of principal and it offers its database as the principal database. If the mirror instance becomes unavailable, the principal continues to offer its database as the principal database.

Synchronous without a witness mode provides a ‘high protection’ operating mode. In this mode if the principal instance becomes unavailable, the database owner can manually failover to the mirror instance. If the mirror instance becomes unavailable, the principal server stops and takes its copy of the database offline.

Asynchronous mode allows for faster processing on the principal instance, but at the cost of not having a perfect copy of the database on the mirror instance. The principal instance commits each new log record locally just before sending the record to the mirror instance. All the while, the mirror instance applies all the outstanding log records to the mirror database in an effort to catch up with the principal database. In this mode performance on the principal is faster. If the mirror instance becomes unavailable, the principal continues to offer its database as the principal database. If the principal becomes unavailable, the mirror stops but is available as a hot standby.

# Transparent client Redirect

The transparent client redirect is a new feature in SQL Server 2005 and works seamlessly with Database Mirroring. This feature is implemented in the new SQL Server 2005 Microsoft Data Access Component (MDAC). The MDAC is aware of both principal and mirror instances. Client application successfully connects to the principal database, and the instance sends back the name of the partner instance, which is then cached on the client. When the connection to principal is lost, MDAC will attempt to re-establish a connection to the principal database. If the attempt fails, MDAC will transparently attempt to establish a connection to the mirror database on the partner instance. If it succeeds, the connection is redirected to the mirror database, which then becomes the new principal database. If the new principal database has a partner instance, then that new partner instance name is retrieved and cached on the client.

# Setting up Database Mirroring

Following steps are required to set up database mirroring session:

1. Identify principal, mirror and optional witness instances
2. Backup and restore the database from the principal to the mirror instance without recovery. The principal database must be in the FULL recovery model before you make the database backup to restore to the mirror server.
3. Make sure the servers involved in the database mirroring session trust each other. Each SQL Server instance login must have rights to connect to the other mirroring server and do its endpoints.
4. Configure Endpoints for partners to communicate. Regardless of the number of mirroring sessions to be supported, the server instance must have only one database mirroring endpoint. If you intend to use a server instance exclusively as a principle/mirror in database mirroring sessions, assign the role of partner to the endpoint (ROLE=PARTNER). If you intend to use a server instance exclusively as a witness in database mirroring sessions, assign the role of witness to the endpoint (ROLE=WITNESS). If you intend to use a server instance as a partner in one or more other database mirroring sessions, assign the role of the endpoint as ALL.

The following example shows the configurations of endpoints for three SQL Server instances in a High Availability mirroring session. The Server instance SQLSRVP is the principal, Server instance SQLSRVM the mirror, and Server instance SQLSRVW the witness.

--Endpoint for initial principal server instance, which

--is the only server instance running on SQLSRVP.

CREATE ENDPOINT endpoint\_mirroring\_name

STATE = STARTED

AS TCP ( LISTENER\_PORT = 5022 )

FOR DATABASE\_MIRRORING (ROLE=PARTNER);

GO

--Endpoint for initial mirror server instance, which

--is the only server instance running on SQLSRVM.

CREATE ENDPOINT endpoint\_mirroring\_name

STATE = STARTED

AS TCP ( LISTENER\_PORT = 5022 )

FOR DATABASE\_MIRRORING (ROLE=PARTNER);

GO

--Endpoint for witness server instance, which

--is the only server instance running on SQLSRVW.

CREATE ENDPOINT endpoint\_mirroring\_name

STATE = STARTED

AS TCP ( LISTENER\_PORT = 5022 )

FOR DATABASE\_MIRRORING (ROLE=WITNESS);

GO

Figure : Endpoints Configuration for Exclusive Role

1. Set up partner database properties for mirroring as follows.

--Set Partner for initial mirror server instance, which

--is the only server instance running on SQLSRVM.

ALTER DATABASE DB\_to\_Mirror

SET PARTNER =

'TCP://SQLSRVP.MYDOMAIN.CORP.MYCOMPANY.COM:5022'

GO

--Set Partner for initial principle server instance, which

--is the only server instance running on SQLSRVP.

ALTER DATABASE DB\_to\_Mirror

SET PARTNER =

'TCP://SQLSRVM.MYDOMAIN.CORP.MYCOMPANY.COM:5022'

GO

--Set Witness for witness server instance, which

--is the only server instance running on SQLRVW.

ALTER DATABASE DB\_to\_Mirror

SET WITNESS =

'TCP://SQLSRVW.MYDOMAIN.CORP.MYCOMPANY.COM:5022'

GOGO

Figure 3: Setup of Partner and Witness database for Mirroring

1. Start database mirroring. In this state the mirror instance sends a request to the principal instance to send all transaction log records that are more recent than latest log record already applied to the mirror database.

The principal sends all of its accumulated log records to the mirror instance. Beginning with the oldest record, the mirror instance applies the received log records to catch up with the principal instance. After sending the accumulated log records, the principal instance sends each subsequent log record immediately. As long as the mirroring session is in effect, the mirror server applies the incoming log records from the principal to the mirror database. The mirror database is continually in recovery, rolling the log forward.