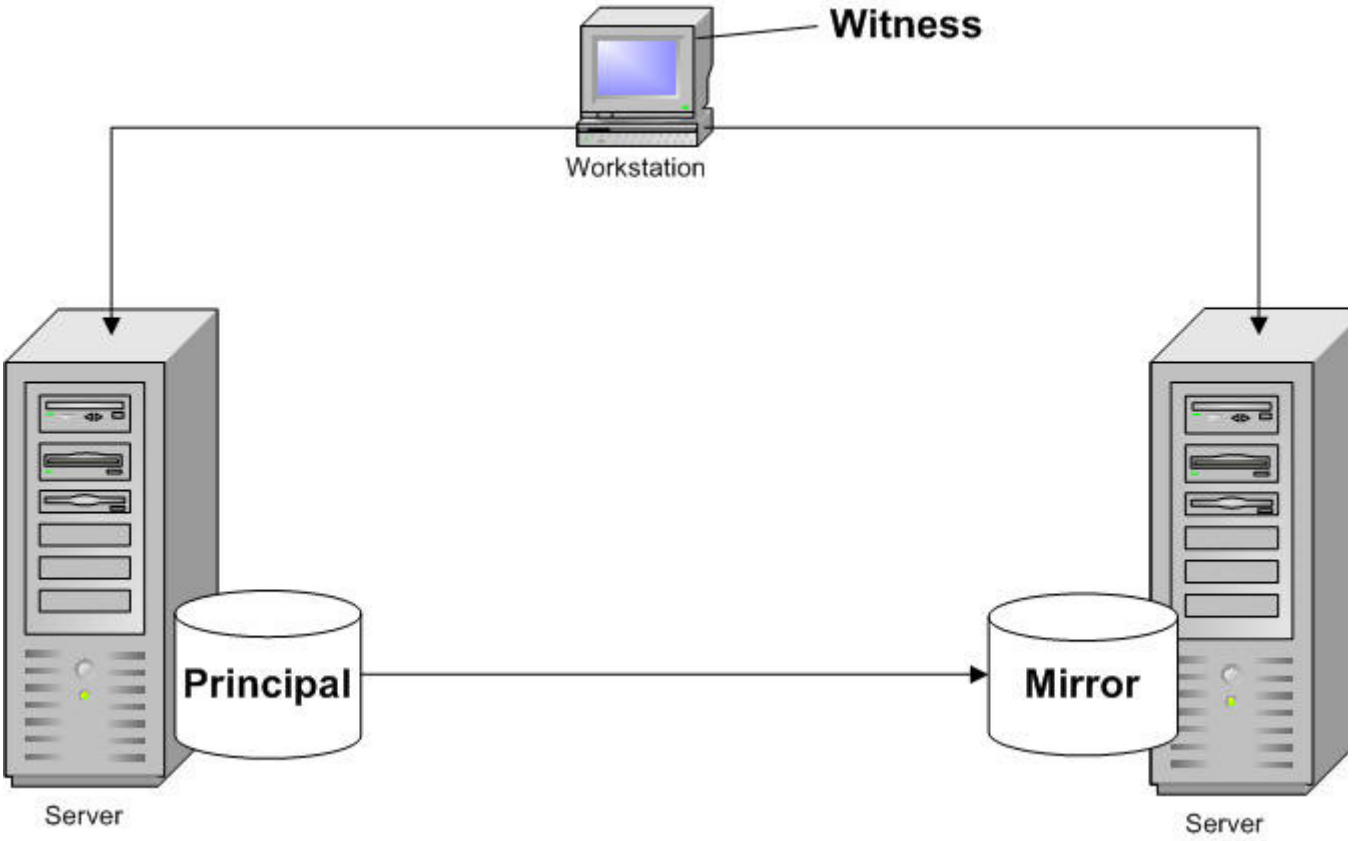
A Look at Database Mirroring

Database Mirroring is a major new technology for achieving high availability in SQL Server 2005. Database Mirroring operates at a database level and is configured on a database-by-database basis. The primary goal of database mirroring is to increase data availability and allow failover in case a server hosting the database becomes unavailable. So including Database Mirroring now we have 4 High Availability in SQL Server 2005 (1) Clustering (2) Log Shipping (3) Replication (Transactional) (4) Mirroring. I can say that Database Mirroring is Log Shipping with automatic failover per database & Clustering on database level without single point of failure without shared disk.



To have successful implementation of Database Mirroring you need to have three instances of SQL Server 2005 installed either Standard or Enterprise Edition for all instances. You must specify the Full recovery model for the each database to participate in database mirroring session. Many applications make use of multiple databases on a single server. One application may reference multiple databases, or perhaps many applications all make use of several databases. However, database mirroring works with a single database at a time. You need to take this into account when designing mirroring into your database architecture.

Database Mirroring comprises of two mandatory roles and a third, optional role

- Principal Role
- Mirror Role
- Witness Server - third and optional role to implement automatic failure detection and failover. SQL Server 2005 Express and Workgroup Edition is a good candidate as it only support role of witness.

Each of the participating servers keeps some metadata about the session and the current state of the databases. You can inspect the session on the principal or mirror server by querying the **sys.database_mirroring** catalog view as shown below. The witness server metadata is returned using a different catalog view:**sys.database_mirroring_witnesses.**

The output of sys.database_mirroring

Partner metadata column	Principal values: Server A	Mirror values: Server B
db_name(database_id)	AdventureWorks	AdventureWorks
mirroring_role_desc	PRINCIPAL	MIRROR
mirroring_safety_level_desc	FULL	FULL
mirroring_state_desc	SYNCHRONIZED	SYNCHRONIZED
mirroring_safety_sequence	1	1
mirroring_role_sequence	1	1
mirroring_partner_instance	TCP://B.corp.XYZ.com:5022	TCP://A. .corp.XYZ.com:5022
mirroring_witness_name	TCP://W.corp.XYZ.com:5022	TCP://W.corp.XYZ.com:5022
mirroring_witness_state_desc	CONNECTED	CONNECTED
mirroring_failover_lsn	95000000007300001	95000000007300001

Before you start mirroring session make sure you initialize each Mirror database to ensure that it is synchronized with the Principal. It means that you need to perform last known full backup of principal database (including, if any, transaction logs backups that have been performed after that backup) and restoring it to Mirror Server with NORECOVERY option. This puts the mirror database in a recovering state and does not allow any connections.

You need to separately transfer logins including SQL Agent jobs and alerts, SQL Server Integration Services packages, support databases, linked server definitions, backup devices, maintenance plans, SQL Mail or Database Mail settings, and possibly Distributed Transaction Coordinator settings, to name a few. SQL Server Integration Services has a Transfer Logins task that you can use to copy logins and passwords from one server to another, but you may still need to set database permissions for those logins. If you transfer logins to a server in a different domain, the SIDs may not match and you will need to match them.

Database Mirroring is configured within SQL Server Management Studio (SSMS) from the database properties, Mirroring page. The connectivity among Principal, Mirror and Witness Servers uses **endpoints**, which is a new feature introduced in SQL Server 2005. Database Mirroring uses TCP

endpoints (with a TCP default port of 5022) for communications with fully qualified computer names and is created with a payload of DATABASE_MIRRORING. Only one endpoint is created for each SQL Server instance.

You can also change the default TCP ports for security reasons. If your principal, mirror, and witness servers reside in the same domain, then use Window based authentication. If they are not in trusted domains, then use Certificate based authentication. You can also encrypt communications between endpoints to enhance security.

The easiest way to set up endpoints is to use the Configure Database Mirroring Security Wizard, which you can invoke by clicking the Configure Security button on the Mirroring page of the Database Properties dialog (Check Books Online for how to create a TCP endpoint).

Database Mirroring can be configured for three different operating modes:

High Availability Operating Mode - This provides durable, synchronous transfer of data between principal and mirror, including automatic failure detection and failover. There is performance overhead on this mode because a transaction is not considered committed until SQL Server has successfully committed it to the transaction log on both the principal and the mirror database. And as the distance between the principal and the mirror increases, the performance impact also increases. There is a continuous ping process between all three to detect failover. If the witness server is not visible from the mirror, you must either reconfigure the operating mode for the database mirroring session or turn off the witness.

Alternatively, you can manually fail over a database mirroring session at the mirror in High Availability Mode by issuing the following command at the principal. You can also issue the same command if you have to take principal down for maintenance.

```
ALTER DATABASE SET PARTNER FAILOVER
```

High Performance Operating Mode - In this configuration you don□t need a WITNESS Server and the Mirror Server acts as an WARM standby and does not support automatic failure detection or failover. There is any asynchronous data transfer between principal and mirror. This mode provide better performance and you can have geographic dispersion between the principal and the mirror. This mode increases latency and can lead to greater data loss in the event of primary database failure.

High Protection Operating Mode - This mode is the same as High Availability Mode except failover is manual and you have to manually promote the mirror to be the principal. Data transfer is synchronous. This one is not the recommended mode and can only be use in the event when you need to replace an existing Witness Server. After replacing or recovering the Witness server, you should change the operating mode back to High Availability.

Database Mirroring Operating Modes

Operating Mode	Transaction Safety	Transfer Mechanism	Quorum Required	Witness Server	Failover Type
High Availability	Full	Synchronous	Yes	Yes	Automatic or Manual
High Protection	Full	Synchronous	Yes	No	Manual Only
High					

Performance	Off	Asynchronous	No	-NA-	Forced Only

Client Connections in Database Mirroring

In SQL Server 2005, if you connect to a database that is being mirrored with ADO.NET or the SQL Native Client, your application can take advantage of the drivers' ability to automatically redirect connections when a database mirroring failover occurs. You must specify the initial principal server and database in the connection string, and optionally the failover partner server.

There are many ways to write the connection string, but here is one example, specifying server A as the principal, server B as the mirror, and AdventureWorks as the database name:

```
"Data Source=A;Failover Partner=B;Initial Catalog=<database name>;Integrated
Security=True;"
```

The failover partner in the connection string is used as an alternate server name if the connection to the initial principal server fails. If the connection to the initial principal server succeeds, then the failover partner name will not be used, but the driver will store the failover partner name that it retrieves from the principal server on the client-side cache.

The great advantage of using the database mirroring support built into ADO.NET and the SQL Native Client driver is that you do not need to recode the application, or place special code in the application, to handle a database mirroring failover.

If you do not use the ADO.NET or SQL Native Client automatic redirection, you can use other techniques that will enable your application to fail over. For example, you could use Network Load Balancing to manually redirect connections from one server to another, while the client just connects to a virtual server name. You might also write your own redirection code and retry logic.

When safety is FULL, a database mirroring session requires a quorum to keep a database in service. A quorum is the minimal relationship among all the connected servers required by a synchronous database mirroring session. Because at least two servers are required for a quorum, when safety is FULL, the principal server must form a quorum with at least one other server to keep the database in service (For further information, see the topic "Quorum in Database Mirroring Sessions" in SQL Server Books Online).

Some tips on hardware level to configure server for principal & mirror.

At the physical server level, you should make sure that the standby server has the same physical CPU and memory configuration, or as close as possible, or else the standby will under perform after a failover. You may also have supporting applications, monitors, or other executables that support the database applications and must be configured on the mirror server.

At the SQL Server level, you should also ensure that the standby has the same SQL Server configurations (for example, AWE, max degree of parallelism). But most fundamental will be the logins and their permissions.

You can initiate a manual failover at the mirror only when the principal is inaccessible and the witness is either off or connected to the mirror. Issue this command at the mirror database.

ALTER DATABASE <database name> SET PARTNER FORCE_SERVICE_ALLOW_DATA_LOSS

The worst scenario will be if the principal server crashes or is damaged. Then you have to remove database mirroring and reinitialize the entire environment. Removing mirroring can be done by right clicking the **<database name>** on the principal, choose properties, in the mirror page, click stop mirroring or issue this command either on principal or mirror.

ALTER DATABASE <database name> SET PARTNER OFF

Conclusion

Database mirroring is a new SQL Server 2005 technology that can deliver high availability and high performance solutions for database redundancy. In database mirroring, transaction log records are sent directly from a principal to a mirror database whenever the principal's transaction log buffer is written to disk. This technique can keep the mirror database nearly up to date with the principal, and with no loss of committed data. In the High Availability operating mode, if the principal fails, the mirror server will automatically become a new principal and recover its database. Database mirroring becomes an important new option in the array of high availability technologies supported by SQL Server 2005.

Database Mirroring Features in the various SQL Server 2005 Editions

Database Mirroring Feature	Enterprise Edition	Developer Edition	Standard Edition	Workgroup Edition	SQL Express
Partner	X	X	X		
Witness	X	X	X	X	X
Safety = FULL	X	X	X		
Safety = OFF	X	X			
Available during UNDO after failover	X	X	X		
Parallel redo	X	X			
Database Snapshots	X	X			