MS SQL Server DBA

Praveen Kumar M
Mb: +91 986 613 0093 (Botim\WhatsApp)
+91 819 729 3434 (WhatsApp)

Mail: praveensqldba12@gmail.com

LinkedIn: https://www.linkedin.com/in/praveenmadupu

Github: https://github.com/praveenmadupu

Youtube: https://www.youtube.com/PraveenMadupu

# DB Mirroring Architecture

## Database Mirroring Architecture in Detail

**Database Mirroring** is a high-availability solution in SQL Server that provides database redundancy by maintaining a copy of a database (the **mirror**) on a different server (the **mirror server**) than the primary server (the **principal**). The principal server hosts the active database, and the mirror server holds the replica database. Changes to the principal database are mirrored to the mirror database in near real-time.

Database mirroring is often used for **high availability**, **disaster recovery**, and **data redundancy**. Although database mirroring has been deprecated in favor of **AlwaysOn Availability Groups**, it is still useful and widely implemented in some environments due to its simplicity and lower resource consumption.

Let's break down the **architecture** of database mirroring in detail.

## Key Components of Database Mirroring

1. **Principal Server (Primary Server)**
    - This is the **active** server where the database is in use.
    - The principal server maintains the **primary copy** of the database that users access.
    - All changes made to the database (insert, update, delete operations) on the principal are logged in the transaction log.
2. **Mirror Server**
    - The mirror server maintains a **restoring** copy of the database.
    - The mirror server's database is synchronized with the principal server's database.
    - The mirror database is in a **restoring** state, meaning it cannot be accessed by users until it is failed over and promoted to the principal role.
    - In some configurations, the mirror can also be **read-only** for reporting, but this depends on the mode used.
3. **Witness Server (Optional)**
    - The **witness server** is an optional server that plays a role in **automatic failover** in high-availability configurations.
    - The witness server doesn't hold a copy of the database but instead helps to monitor the status of the principal and mirror servers to determine when a failover is necessary.
    - The witness server allows for **automatic failover** in a **high-safety mode** but is not required for basic mirroring.

## Modes of Database Mirroring

Database mirroring operates in different modes, each offering different features and behaviors. The modes determine how the transaction log is applied to the mirror database, the level of data protection, and whether or not automatic failover is supported.

1. **High Availability Mode (High-Safety Mode)**
    - **Data Safety**: This mode provides the highest level of data safety, as transactions are written to both the principal and mirror databases synchronously.
    - In high availability mode, the principal and mirror databases are synchronized in real-time. If the principal database goes down, the mirror database can be quickly promoted to the principal role, minimizing data loss.
    - **Automatic Failover**: When a **witness server** is configured, automatic failover can occur. If the principal server fails, the mirror server automatically takes over as the new principal.
    - **Performance Impact**: This mode may have some performance overhead because the system waits for

the transaction log to be committed to both the principal and mirror databases before the transaction is completed.

2. **High Performance Mode (Asynchronous Mode)**

    - **Data Safety**: In this mode, the transaction logs are **asynchronously** sent to the mirror server. This means that the principal server does not wait for the transaction log to be applied on the mirror server before committing the transaction to the principal database.
    - **Automatic Failover**: High performance mode does **not support automatic failover**, as the mirror database may lag behind the principal server.
    - **Performance Impact**: There is little or no performance impact on the principal server, as there is no waiting for the mirror server to acknowledge receipt of the transaction log.

3. **High-Safety with No Witness (Manual Failover)**

    - This mode is similar to high availability mode, but it does not have an automatic failover mechanism. If the principal server fails, a manual intervention is needed to promote the mirror server to become the new principal.

## Database Mirroring Workflow

The core process of database mirroring revolves around the continuous replication of transaction log entries from the principal server to the mirror server. Below is a step-by-step breakdown of the workflow:

1. **Transaction Log Generation (Principal Server)**

    - When any operation (insert, update, delete) occurs on the principal database, SQL Server writes the changes to the **transaction log** of the database.
    - These changes are immediately captured in the transaction log.

2. **Transaction Log Sent to Mirror Server**

    - The transaction log from the principal server is sent to the mirror server, where it is stored temporarily.
    - The transaction log is transferred over the network, and it will be either synchronously (in high-safety mode) or asynchronously (in high-performance mode) applied to the mirror database.

3. **Restoring the Transaction Log (Mirror Server)**

    - The mirror server receives the transaction log and **restores** it on the mirror database.
    - The mirror database is typically in the **restoring** state, meaning no user can access it, and it waits for the next transaction log to be applied.

4. **Commit or Acknowledge**

    - In **high-safety mode**, once the transaction is committed on the principal, the system waits for an acknowledgment from the mirror server that it has successfully written the transaction log before completing the operation.
    - In **high-performance mode**, the transaction is committed on the principal without waiting for the mirror server to acknowledge.

## Failover in Database Mirroring

Failover in database mirroring refers to the process of switching the role of the principal and mirror databases in case of a failure or for planned maintenance.

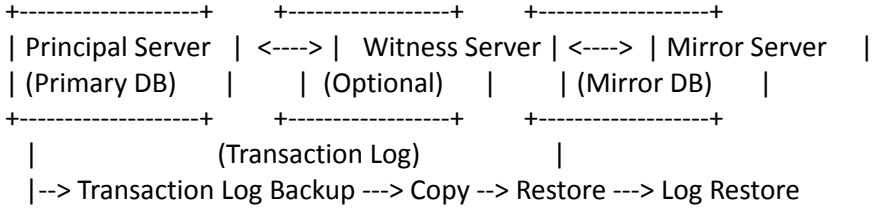
1. **Manual Failover**

    - In manual failover, an administrator can force a role reversal. This is done when the principal server is operational, but the administrator needs to switch the roles, often for maintenance or testing

purposes.
    - The failover is initiated by running the **ALTER DATABASE** command in SQL Server or through SQL Server Management Studio (SSMS).
2. **Automatic Failover (High-Safety Mode with Witness)**
    - When a **witness server** is configured in **high-safety mode**, SQL Server can perform an **automatic failover** when the principal server fails.
    - The witness server constantly monitors the status of the principal and mirror servers and helps to determine when a failover should occur.
    - When the principal server becomes unavailable, the mirror server is automatically promoted to the principal role, and the original principal is demoted to the mirror role.
3. **Role Switching after Failover**
    - After failover, the new principal database takes over operations, and the former principal database becomes the mirror.
    - The transaction log continues to be sent from the new principal to the mirror.

## Database Mirroring Architecture Diagram

Below is a simple architecture diagram illustrating how database mirroring works:




## Advantages of Database Mirroring

1. **High Availability**:
    - Database mirroring provides high availability by maintaining a real-time copy of the database, which can be quickly brought online if the principal database fails.
2. **Data Protection**:
    - By synchronizing transaction logs between the principal and mirror databases, mirroring ensures that the mirror database is a near-identical copy, protecting against data loss.
3. **Automatic Failover**:
    - In **high-safety mode with a witness**, mirroring supports **automatic failover**, providing minimal downtime in case of failure.
4. **Simple Setup**:
    - Database mirroring is relatively easy to set up compared to other high-availability solutions like **AlwaysOn Availability Groups** or **Replication**.
5. **Low Overhead**:
    - For read-heavy environments, the mirror can be used for reporting purposes, offloading the principal server from heavy read operations.

## Disadvantages of Database Mirroring

1. **No Automatic Failover without Witness**:
    - If no witness server is configured, failover is **manual**, which means a DBA must intervene to bring the mirror database online in the event of a failure.
2. **No Shared Storage**:
    - Unlike some clustering solutions, database mirroring does not use shared storage, meaning each server requires its own local storage.
3. **Performance Impact in High-Safety Mode**:
    - In **high-safety mode**, where transaction logs are written synchronously to both the principal and mirror servers, there can be a performance overhead due to the wait for acknowledgment from the mirror server.
4. **Limited Scalability**:
    - Unlike **AlwaysOn Availability Groups**, database mirroring is typically **1-to-1**. There is no direct support for multiple mirrors or additional replicas.
5. **Deprecation**:
    - **Database mirroring** has been deprecated in SQL Server 2012 in favor of **AlwaysOn Availability Groups**, which offer more features, including support for read-only replicas and multi-database availability groups.

## Summary:

**Database mirroring** is a simple yet effective high-availability solution that keeps databases synchronized between a primary server (principal) and a secondary server (mirror). It provides data protection and quick failover in case of server failures. Although it has been largely superseded by **AlwaysOn Availability Groups** in newer versions of SQL Server, database mirroring remains a viable solution for organizations with simple availability requirements, especially in environments where synchronous replication of a single database is required.

For organizations with complex availability or disaster recovery needs, **AlwaysOn** may offer additional benefits, but database mirroring still serves as an effective, less resource-intensive solution for many use cases.