https://www.sqldbachamps.com

Praveen Madupu +91 98661 30093
Sr SQL Server DBA, Dubai
praveensqldba12@gmail.com

Database Mirroring in SQL Server is a high-availability solution that provides redundancy by maintaining a mirror copy of a database on a different server. However, several issues may arise during its configuration or operation that can impact its performance or reliability.

Below are common issues with SQL Server Database Mirroring and their detailed resolutions.

1. Database Mirroring Session Won't Start

Issue: The mirroring session doesn't start after configuration, and the database remains in "Restoring..." state.

Causes:

- Missing or incomplete backup/restore process on the mirror server.
- Network configuration issues between the principal and mirror servers.
- Database versions or editions are incompatible (e.g., mismatched SQL Server versions).

Resolution:

- **Restore Database with NORECOVERY**: Ensure that the mirrored database was restored with the `NORECOVERY` option on the mirror server. If it was not, the mirroring session cannot start. Use:

```
RESTORE DATABASE [DatabaseName] FROM DISK = 'path_to_backup' WITH NORECOVERY;
```

- Also, restore all necessary transaction log backups in `NORECOVERY` mode before setting up mirroring.
- **Network Configuration**: Verify that both the principal and mirror servers can communicate over the configured endpoints (check firewalls, network ports, and DNS resolution). Use `telnet` to test connectivity on the mirroring ports (default 5022).
- **SQL Server Version Compatibility**: Ensure that both the principal and mirror servers are running compatible versions and editions of SQL Server. Database Mirroring requires the same SQL Server edition (Standard or Enterprise) on both servers for synchronous mirroring. Upgrade the SQL Server version if needed.

Verification:

- Check SQL Server Error Logs and the mirroring monitor for error messages.
- Ensure the mirroring endpoints are configured correctly using:

```
SELECT * FROM sys.database_mirroring_endpoints;
```

Praveen Madupu +91 98661 30093
Sr SQL Server DBA, Dubai
praveensqldba12@gmail.com

2. Mirroring State Stuck in "SUSPENDED"

Issue: The mirroring session goes into a "SUSPENDED" state, stopping synchronization between the principal and mirror databases.

Causes:

- The mirror server is unavailable, or there are network issues.
- Transaction log file on the principal is growing due to unsent logs.
- Manual suspension of the mirroring session (intentional or accidental).

Resolution:

- **Check Network and Mirror Availability**: Ensure that the mirror server is online and reachable over the network. Verify that SQL Server instances are running on both the principal and mirror servers. Use network troubleshooting tools like `ping` and `telnet` to check connectivity.
- **Unsuspend Mirroring**: If the mirroring session was manually suspended, use the following command to resume it:

  ALTER DATABASE [DatabaseName] SET PARTNER RESUME;

- **Resolve Unsent Log Issue**: Check the transaction log on the principal server to ensure it isn't growing due to unsent logs. If the logs are large, you may need to manually resume the session and monitor the sync process.

Verification:

- Use the following query to check the mirroring state:

  SELECT database_id, mirroring_state_desc FROM sys.database_mirroring;

- Ensure that the `mirroring_state_desc` shows **"SYNCHRONIZED"** or **"SYNCHRONIZING"** after the issue is resolved.

3. Database Mirroring Endpoint Issues

Issue: The Database Mirroring endpoint is not functioning correctly, causing mirroring to fail.

Causes:

- Endpoint not started or improperly configured.
- Port conflicts or firewall restrictions.
- Authentication/permissions issues on the endpoints.

Resolution:

- **Start Endpoint**: Ensure the Database Mirroring endpoints are created and running on both the principal and mirror servers. Use the following command to start the endpoint if it is stopped:

Praveen Madupu +91 98661 30093
Sr SQL Server DBA, Dubai
praveensqldba12@gmail.com

ALTER ENDPOINT [MirroringEndpointName] STATE = STARTED;

Check Endpoint Configuration: Verify the configuration of the mirroring endpoint on both servers. Ensure they are listening on the correct port and have matching encryption and authentication settings. Use the query:

SELECT * FROM sys.database_mirroring_endpoints;

- **Firewall Configuration**: Ensure that the port used by the mirroring endpoint (default 5022) is open on both the principal and mirror servers' firewalls. Use `telnet` or a similar tool to test the connection between servers on this port.
- **Authentication**: If the endpoints use certificates for authentication, ensure that the certificates are properly set up and valid on both servers.

Verification:

- Review the SQL Server error log for endpoint-related errors.
- Use the following query to verify endpoint status:

SELECT state_desc, port FROM sys.database_mirroring_endpoints;

4. Mirroring State in "DISCONNECTED"

Issue: The database mirroring session is in a "DISCONNECTED" state, indicating a communication problem between the principal and mirror servers.

Causes:

- Temporary network issues or network partition.
- Misconfigured network endpoints (e.g., wrong IP, port settings).
- Resource constraints on either the principal or mirror server (e.g., high CPU or memory usage).

Resolution:

- **Check Network and Reconnect**: If the disconnection is caused by a temporary network issue, verify the connection between the principal and mirror servers. After the network issue is resolved, SQL Server should automatically reconnect. If it doesn't, you may need to restart the mirroring session using:

  ALTER DATABASE [DatabaseName] SET PARTNER RESUME;

- **Verify Endpoints**: Confirm that the mirroring endpoints on both servers are correct and match in terms of IP address, port, and authentication. Also, ensure that there are no port conflicts (e.g., other services using the same port).
- **Check Resource Usage**: Ensure that both the principal and mirror servers have adequate system resources (CPU, memory, and disk space). Performance bottlenecks can cause disconnection issues in high-traffic environments.

Praveen Madupu +91 98661 30093
Sr SQL Server DBA, Dubai
praveensqldba12@gmail.com

Verification:

- Check mirroring session status using:

  SELECT database_id, mirroring_state_desc FROM sys.database_mirroring;

- Review SQL Server and Windows Event logs for any disconnection-related errors.

5. Mirroring in "SYNCHRONIZING" State for a Long Time

Issue: The database mirroring session stays in "SYNCHRONIZING" mode for an extended period without becoming "SYNCHRONIZED".

Causes:

- High transaction log activity on the principal database, leading to a backlog of unsent transaction logs.
- Bandwidth limitations or network latency between the principal and mirror servers.
- Insufficient disk I/O performance on the mirror server.

Resolution:

- **Check Transaction Log Growth**: Monitor the transaction log on the principal server using `DBCC SQLPERF(LOGSPACE)` to see if it's growing rapidly. Consider adjusting your transaction log backup strategy to reduce the size of the logs sent for mirroring.
- **Optimize Network Bandwidth**: Check the network bandwidth between the principal and mirror servers. If possible, increase the available bandwidth or use compression to reduce the size of the data being mirrored.
- **Improve Disk I/O**: Ensure that the mirror server has sufficient disk I/O capacity to handle the incoming transaction logs. Consider upgrading the storage or optimizing existing disk performance.

Verification:

- Monitor the mirroring state using:

SELECT database_id, mirroring_state_desc FROM sys.database_mirroring;

Check the amount of unsent transaction logs using the following query:

SELECT DB_NAME(database_id), log_send_queue_size, redo_queue_size

FROM sys.dm_db_mirroring_connections;

Praveen Madupu +91 98661 30093
Sr SQL Server DBA, Dubai
praveensqldba12@gmail.com

6. Database Mirroring Security and Authentication Failures

Issue: Database mirroring fails due to security or authentication issues, such as mismatched logins or endpoint permissions.

Causes:

- Incorrect service account configuration or insufficient permissions for the SQL Server instances.
- Mismatched or missing certificates when using certificate-based authentication.

Resolution:

- **Correct Service Accounts**: Ensure that the SQL Server service accounts on both the principal and mirror servers have appropriate permissions to communicate with each other. These accounts must have **CONNECT** permissions to the database mirroring endpoint on the other server. Use:

  ```
  GRANT CONNECT ON ENDPOINT::[MirroringEndpointName] TO [Domain\User];
  ```

- **Check Certificates**: If using certificate-based authentication, ensure the certificates on both the principal and mirror servers are valid, match each other, and haven't expired. Recreate or reissue the certificates if necessary.

Verification:

- Review SQL Server Error Logs for security-related error messages.
- Use the following query to check endpoint permissions:

```
SELECT e.name, p.grantee_principal_id, p.permission_name

FROM sys.database_mirroring_endpoints AS e

JOIN sys.server_permissions AS p

ON e.endpoint_id = p.major_id;
```

7. Automatic Failover Not Occurring (High-Safety with Witness)

Issue: Automatic failover does not happen when using High-Safety Mode with Witness, even though the principal server has failed.

Causes:

- The witness server is not properly configured or is unavailable.
- Network issues between the witness server and either the principal or mirror server.
- Insufficient quorum due to network partitioning.

Praveen Madupu +91 98661 30093
Sr SQL Server DBA, Dubai
praveensqldba12@gmail.com

Resolution:

- **Verify Witness Configuration**: Ensure that the witness server is correctly configured and that both the principal and mirror servers can communicate with it. Use the following command to check the witness:

  ```
  SELECT name, mirroring_witness_name FROM sys.database_mirroring;
  ```

- **Check Quorum**: For automatic failover, a quorum (witness and at least one partner) must be available. Ensure that both the witness and mirror servers are online and reachable. Resolve any network issues causing partitioning.
- **Network Configuration**: Verify the network configuration and endpoints to ensure that the witness can communicate with both the principal and mirror servers. Check the firewall rules and network latency.

Verification:

- Use the **Database Mirroring Monitor** in SQL Server Management Studio (SSMS) to review the status of the witness and automatic failover configuration.
- Check SQL Server Error Logs for messages related to witness communication.

Best Practices for Database Mirroring:

1. **Monitor Regularly**: Use Database Mirroring Monitor in SSMS or custom scripts to monitor the health of the mirroring session, especially in a production environment.
2. **Test Failover**: Periodically test manual and automatic failover scenarios to ensure the mirroring setup functions as expected during an actual failure.
3. **Optimize Log Backup Strategy**: Regular transaction log backups on the principal database can help reduce the transaction log size and avoid large backlogs in synchronous mirroring.
4. **Network Optimization**: Ensure low-latency, high-bandwidth network connections between the principal, mirror, and witness servers to reduce synchronization issues.
5. **Use Witness with Caution**: In production environments with High-Safety Mode, the witness can cause unnecessary failovers in certain scenarios. Monitor its performance closely.

By following these resolutions and best practices, you can troubleshoot and optimize SQL Server Database Mirroring effectively.