**Database Snapshots in Database Mirroring** in SQL Server are a feature that allows you to create a read-only copy of a database at a specific point in time. This copy is useful for reporting, backup, or data recovery purposes. In a **Database Mirroring** environment, the primary database and the mirrored database work together to provide high availability and disaster recovery. However, using **Database Snapshots** in a mirrored environment can be particularly useful for creating point-in-time views of the database without impacting the primary or mirrored databases.

Let's break down how **DB Snapshots** work in **Database Mirroring** in SQL Server and how they are used.

### **Key Concepts:**

1. **Database Mirroring**: Database mirroring in SQL Server involves two copies of a database: one **primary** database and one **mirrored** database. The primary database is the live database that handles all transactions, and the mirrored database is a copy of the primary database maintained in real-time. In case of failure of the primary database, the mirrored database can be automatically or manually promoted to the primary role.
2. **Database Snapshot**: A **Database Snapshot** is a read-only, point-in-time copy of a database. A snapshot is created by SQL Server using a sparse file that stores only the changes made to the database after the snapshot was taken. Snapshots are useful for backup, reporting, and testing because they allow for querying historical data without affecting the original database.

### **Using Database Snapshots in Database Mirroring**

In a **mirrored database setup**, you can use database snapshots for the **primary database** to facilitate reporting, testing, or backup operations, while keeping the mirrored database available for high availability. However, creating a snapshot on the **mirrored database** itself is **not supported** while database mirroring is enabled. This is because the mirrored database is always in a **restoring** state and can't be used for snapshots or other operations like backups.

### **Key Points to Remember:**

1. **Snapshots can only be created on the primary database** in a mirrored environment.
2. **Snapshots are read-only**: Once a snapshot is created, you cannot make any changes to it.
3. **Space Efficiency**: The snapshot only stores changes after the snapshot is created, making it relatively space-efficient.
4. **Use Cases**:
   * **Reporting**: You can create snapshots of the primary database to run reports without affecting the performance of the live database.
   * **Testing**: You can use snapshots to test certain operations or restore states without impacting production.
   * **Backup**: Snapshots can be used as a backup source if the original database is being updated frequently.

### **Example of Database Snapshot in a Database Mirroring Environment:**

Let's go through a step-by-step example of how you can create a **Database Snapshot** on the **Primary** database in a mirrored setup.

#### **Step 1: Setup Database Mirroring**

Assuming you already have database mirroring set up between the primary and mirrored databases. Here is an example of how database mirroring might be configured:

**-- On the Primary Server**

CREATE DATABASE MyDatabase

ON (NAME = MyDatabase\_data, FILENAME = 'C:\MyDatabase\MyDatabase.mdf')

LOG ON (NAME = MyDatabase\_log, FILENAME = 'C:\MyDatabase\MyDatabase\_log.ldf');

GO

**-- On the Mirror Server**

-- Restore the database from the primary server with NORECOVERY

RESTORE DATABASE MyDatabase

FROM DISK = 'C:\Backup\MyDatabase.bak'

WITH NORECOVERY;

GO

This sets up a **primary** and **mirrored** database pair.

#### **Step 2: Create a Database Snapshot on the Primary Database**

Once the database mirroring is set up and running, you can create a snapshot of the primary database. Here's how to create a snapshot:

-- Create a snapshot of the primary database

CREATE DATABASE MyDatabase\_Snapshot

ON

(

NAME = MyDatabase\_data,

FILENAME = 'C:\MyDatabase\MyDatabase\_Snapshot.ss'

)

AS SNAPSHOT OF MyDatabase;

GO

**Explanation**:

* MyDatabase\_Snapshot: This is the name of the snapshot.
* The snapshot is created on the primary database, using the primary database's file as the base.
* The snapshot is created using the AS SNAPSHOT OF clause, which points to the original database (MyDatabase in this case).

#### **Step 3: Query the Snapshot**

You can now query the **snapshot** database like a regular database, but it is read-only. For example:

-- Querying the snapshot database

USE MyDatabase\_Snapshot;

SELECT \* FROM SomeTable;

GO

You can use this snapshot for reporting, running read-only queries, or other non-modifying operations.

#### **Step 4: Drop the Snapshot**

Once you no longer need the snapshot, you can drop it:

-- Drop the snapshot once it's no longer needed

DROP DATABASE MyDatabase\_Snapshot;

GO

### **How the Snapshot Works in a Mirrored Setup**

* **Primary Database**: The snapshot is created on the **primary database**. Any data changes made on the primary database after the snapshot creation will be tracked in the snapshot file.
* **Mirrored Database**: The mirrored database remains in a read-write **restoring** state, so snapshots cannot be taken directly on it.

### **Limitations and Considerations:**

1. **Snapshots on the Primary Only**: As mentioned, snapshots can only be created on the **primary database**. You cannot create a snapshot on the mirrored database since it is in the RESTORING state.
2. **Performance Overhead**: Snapshots do come with performance overhead. Since the snapshot only stores changes to the database after it was created, this can increase disk space usage, especially in write-intensive systems.
3. **Mirror Failover**: If a failover occurs and the **mirror** becomes the primary database, you'll need to recreate the snapshot on the new **primary** database if you need one.
4. **Backup and Recovery**: While snapshots are read-only, they are not intended as a replacement for traditional backups. Snapshots are more suited for reporting and quick recovery from a known point in time for certain scenarios. Always perform regular backups of your primary database.

### **Summary:**

In SQL Server, **Database Snapshots** in the context of **Database Mirroring** allow you to create point-in-time views of your primary database without impacting its operations, but only for the **primary database**. Snapshots are primarily useful for read-only purposes like reporting and testing. They cannot be created on the mirrored database due to its RESTORING state.

By creating snapshots on the primary database, you can run reporting queries without affecting the performance of the primary database or mirror, ensuring that your high-availability setup is not disrupted by backup or reporting activities.