SQL Server Database Backups and Restores

Backup/Restore Type	Description	Purpose	Backup Components	Restore Scenarios	Considerations
Full Backup	Captures the entire database, including the transaction log, at a specific point in time.	- Provides a complete snapshot of the database for recovery.	- All data and objects within the database. - Part of the transaction log to ensure consistency.	- Restore the entire database to the state when the full backup was taken.	- Should be performed regularly. - Forms the base for all other backup types (differential, log). - Can be time-consuming for large databases.
Differential Backup	Captures only the data that has changed since the last full backup.	- Reduces the amount of data to back up compared to a full backup. - Faster to execute and restore.	- All changes made since the last full backup. - Does not include unmodified data.	- Requires the last full backup and the most recent differential backup for restoration.	- Should be used in conjunction with full backups. - Multiple differential backups can be restored sequentially if needed. - Faster than full backups.

Transaction Log Backup	Captures all transactions since the last log backup or since the start of the full backup.	- Allows point-in-time recovery. - Truncates the transaction log to free up space.	- All transactions that occurred since the last log backup.	- Requires the full backup, the latest differential backup (if applicable), and all subsequent log backups for restoration.	- Essential for Full and Bulk-Logged recovery models. - Should be performed frequently to prevent the transaction log from growing too large.
Copy-Only Backup	Creates a backup that does not affect the sequence of regular backups.	- Useful for ad-hoc backups without disrupting the backup strategy. - Does not reset the differential backup base.	- A complete snapshot like a full backup. - Does not interfere with regular backup cycles.	- Restore using the copy-only backup independently of the regular backup schedule.	- Does not interfere with the backup sequence. - Useful for temporary backups, e.g., before an upgrade or migration.
Tail-Log Backup	Captures the transaction log at the end of a restore sequence, typically before a database is lost or fails.	- Ensures no data loss by capturing all transactions up to the point of failure. - Required for recovery after a disaster.	- The tail end of the transaction log.	- Restore as part of a point-in-time recovery. - Typically used in disaster recovery scenarios.	- Must be taken before restoring a database to avoid losing uncommitted transactions. - Typically the last step before a restore operation.

Restore Options:

Restore with Recovery	Finalizes the restore operation and makes the database operational.	- Ensures the database is ready for use after restoration. - Applies any pending transactions.	- Applies transaction log records and commits transactions.	- Restore the database and make it operational for users. - No further restores are possible after this operation.	- Typically the last step in a restore sequence. - Cannot apply additional transaction log backups after this step.
Restore with NoRecovery	Prepares the database for additional restore operations but keeps it non-operational.	- Allows applying additional backups (e.g., log backups) before making the database operational.	- Leaves the database in a restoring state, awaiting further backup restores.	- Used in multi-step restore processes, such as restoring full, differential, and transaction log backups.	- Must be used when planning to restore additional backups. - The database remains unavailable until the final restore with recovery is performed.
Restore with Standby	Restores the database in read-only mode, allowing further backups to be restored.	- Useful for reporting or auditing while the database is still in recovery mode.	- Leaves the database in a read-only state after applying the current backup.	- Used when the database needs to be accessed (read-only) between restore operations.	- Allows querying the database in standby mode. - The database can be brought online fully after the final restore.

Key Points:

- **Full Backups** are the foundation of any backup strategy and must be taken regularly to ensure complete data protection.
- **Differential Backups** complement full backups by only capturing changes since the last full backup, making them faster to execute and restore.
- **Transaction Log Backups** are crucial for point-in-time recovery and managing transaction log size in Full and Bulk-Logged recovery models.

- **Copy-Only Backups** are ideal for ad-hoc backups that do not interfere with the regular backup sequence.
- **Tail-Log Backups** are critical in disaster recovery scenarios to ensure no data is lost during the final stages of a restore operation.
- **Restore Options (Recovery, NoRecovery, Standby)** provide flexibility in managing the database's state during and after restore operations, depending on the recovery scenario.