Selecting a Recovery Model

Microsoft® SQL Server™ provides three recovery models to:

- Simplify recovery planning.
- Simplify backup and recovery procedures.
- Clarify tradeoffs between system operational requirements.

These models each address different needs for performance, disk and tape space, and protection against data loss. For example, when you choose a recovery model, you must consider the tradeoffs between the following business requirements:

- Performance of large-scale operation (for example, index creation or bulk loads).
- Data loss exposure (for example, the loss of committed transactions).
- Transaction log space consumption.
- Simplicity of backup and recovery procedures.

Depending on what operations you are performing, more than one model may be appropriate. After you have chosen a recovery model or models, plan the required backup and recovery procedures.

This table provides an overview of the benefits and implications of the three recovery models.

<table>
<thead>
<tr>
<th>Recovery model</th>
<th>Benefits</th>
<th>Work loss exposure</th>
<th>Recover to point in time?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>Permits high-performance bulk copy operations. Reclaims log space to keep space requirements small.</td>
<td>Changes since the most recent database or differential backup must be redone.</td>
<td>Can recover to the end of any backup. Then changes must be redone.</td>
</tr>
<tr>
<td>Full</td>
<td>No work is lost due to a lost or damaged data file. Can recover to an arbitrary point in time (for example, prior to application or user error).</td>
<td>Normally none. If the log is damaged, changes since the most recent log backup must be redone.</td>
<td>Can recover to any point in time.</td>
</tr>
<tr>
<td>Bulk-Logged</td>
<td>Permits high-performance bulk copy operations. Minimal log space is used by bulk operations.</td>
<td>If the log is damaged, or bulk operations occurred since the most recent log backup, changes since that last backup must be redone. Otherwise, no work is lost.</td>
<td>Can recover to the end of any backup. Then changes must be redone.</td>
</tr>
</tbody>
</table>
**Simple Recovery**

Simple Recovery requires the least administration. In the Simple Recovery model, data is recoverable only to the most recent full database or differential backup. Transaction log backups are not used, and minimal transaction log space is used. After the log space is no longer needed for recovery from server failure, it is reused.

The Simple Recovery model is easier to manage than the Full or Bulk-Logged models, but at the expense of higher data loss exposure if a data file is damaged.

**Important:** Simple Recovery is not an appropriate choice for production systems where loss of recent changes is unacceptable.

When using Simple Recovery, the backup interval should be long enough to keep the backup overhead from affecting production work, yet short enough to prevent the loss of significant amounts of data.

**Full and Bulk-Logged Recovery**

Full Recovery and Bulk-Logged Recovery models provide the greatest protection for data. These models rely on the transaction log to provide full recoverability and to prevent work loss in the broadest range of failure scenarios.

The Full Recovery model provides the most flexibility for recovering databases to an earlier point in time.

The Bulk-Logged model provides higher performance and lower log space consumption for certain large-scale operations (for example, create index or bulk copy). It does this at the expense of some flexibility of point-in-time recovery.

Because many databases undergo periods of bulk loading or index creation, you may want to switch between Bulk-Logged and Full Recovery models.

**Simple Recovery**

With the Simple Recovery model, the database can be recovered to the point of the last backup. However, you cannot restore the database to the point of failure or to a specific point in time. To do that, choose either the Full Recovery or Bulk-Logged Recovery model.

The backup strategy for simple recovery consists of:

- Database backups.
- Differential backups (optional).

To recover in the event of media failure

1. Restore the most recent full database backup.
2. If differential backups exist, restore the most recent one.

Changes since the last database or differential backup are lost.
Full Recovery

The Full Recovery model uses database backups and transaction log backups to provide complete protection against media failure. If one or more data files is damaged, media recovery can restore all committed transactions. In-process transactions are rolled back.

Full Recovery provides the ability to recover the database to the point of failure or to a specific point in time. To guarantee this degree of recoverability, all operations, including bulk operations such as SELECT INTO, CREATE INDEX, and bulk loading data, are fully logged.

The backup strategy for full recovery consists of:

- Database backups.
- Differential backups (optional).
- Transaction log backups.
- Full and bulk-logged recovery are similar and many users of the Full Recovery model will use the Bulk-Logged model on occasion.

Recovering in the Event of Media Failure

You can restore a database to the state it was in at the point of failure if the current transaction log file for the database is available and undamaged. To restore the database to the point of failure:

1. Back up the currently active transaction log.
2. Restore the most recent database backup without recovering the database.
3. If differential backups exist, restore the most recent one.
4. Restore each transaction log backup created since the database or differential backup in the same sequence in which they were created without recovering the database.
5. Apply the most recent log backup (created in Step 1), and recover the database.

Important: To protect against loss of transactions under the Full Recovery model, the transaction log must be protected against damage. It is strongly recommended that fault-tolerant disk storage be used for the transaction log.

Bulk-Logged Recovery

The Bulk-Logged Recovery model provides protection against media failure combined with the best performance and minimal log space usage for certain large-scale or bulk copy operations. These operations are minimally logged:

- SELECT INTO.
- Bulk load operations (bcp and BULK INSERT).
- CREATE INDEX (including indexed views).
- text and image operations (WRITETEXT and UPDATETEXT).

In a Bulk-Logged Recovery model, the data loss exposure for these bulk copy operations is greater than in the Full Recovery model. While the bulk copy operations are fully logged under the Full Recovery model, they are minimally logged and cannot be controlled on an operation-by-operation basis under the Bulk-Logged Recovery model. Under the Bulk-Logged Recovery model, a damaged data file can result in having to redo work manually.
In addition, the Bulk-Logged Recovery model only allows the database to be recovered to the end of a transaction log backup when the log backup contains bulk changes. Point-in-time recovery is not supported.

The backup strategy for bulk-logged recovery consists of:

- Database backups.
- Differential backups (optional).
- Log backups.

Backing up a log that contains bulk-logged operations requires access to all data files in the database. If the data files are not accessible, the final transaction log cannot be backed up and all committed operations in that log will be lost.

**To recover in the event of media failure**

1. Back up the currently active transaction log.
2. Restore the most recent full database backup.
3. If differential backups exist, restore the most recent one.
4. Apply in sequence all transaction log backups created since the most recent differential or full database backup.
5. Manually redo all changes since the most recent log backup.

**Important:** If the active transaction log is lost (for example, due to hardware failure on the disk containing the transaction log files), all transactions in that log are lost. To prevent loss of the active transaction log, place the transaction log files on mirrored disks.