

The background features a complex diagram illustrating database backup and recovery concepts. It includes a timeline with various backup types: a large cylinder for 'Full Backup' at the start, followed by a bracketed group of six smaller cylinders labeled 'Transaction Log Backups' with a time range of '01:00 - 06:00'. This is followed by a 'Differential Backup' cylinder at '06:00'. Another bracketed group of four cylinders labeled 'Transaction Log Backups' follows, with a time range of '07:00 - 12:00'. Then, another 'Differential Backup' cylinder is shown at '12:00'. A final 'Transaction Log Backup' cylinder is at '13:00'. A red dashed line with an arrow points from the text 'Some crucial data were deleted at 13:30' to a bomb icon at '13:30'. A bracket at the bottom right indicates a 'Log' period starting at '13:30'. In the center, there is a server rack icon.

# Database Backup and Recovery Concepts

A Comprehensive Guide

# Backup Types

## Full vs. incremental vs. differential backup

### Full backup

Data is copied in its entirety every time.



BACKUP REPOSITORY



1<sup>st</sup> backup



2<sup>nd</sup> backup



3<sup>rd</sup> backup



4<sup>th</sup> backup

### Incremental backup

Data is copied in its entirety to begin with, and then only new or updated data is backed up each time a backup is initiated after that.



BACKUP REPOSITORY



1<sup>st</sup> backup



2<sup>nd</sup> backup



3<sup>rd</sup> backup



4<sup>th</sup> backup

### Differential backup

Data is copied in its entirety to begin with, and then subsequent backups copy any data that has changed since the last full backup.



BACKUP REPOSITORY



1<sup>st</sup> backup



2<sup>nd</sup> backup



3<sup>rd</sup> backup



4<sup>th</sup> backup

## Full Backup

Complete copy of all data

### ✓ Pros

Quick restore, easy storage management

### ✗ Cons

Most storage space, longest backup time

**Best for:** *Small businesses with small data volumes*

## + Incremental Backup

Backs up files changed since the last backup

### ✓ Pros

Efficient storage use, fast backups

### ✗ Cons

Time-consuming restoration, requires all backup files

**Best for:** *Businesses with large data volumes*

## + Differential Backup

Backs up files changed since the last full backup

### ✓ Pros

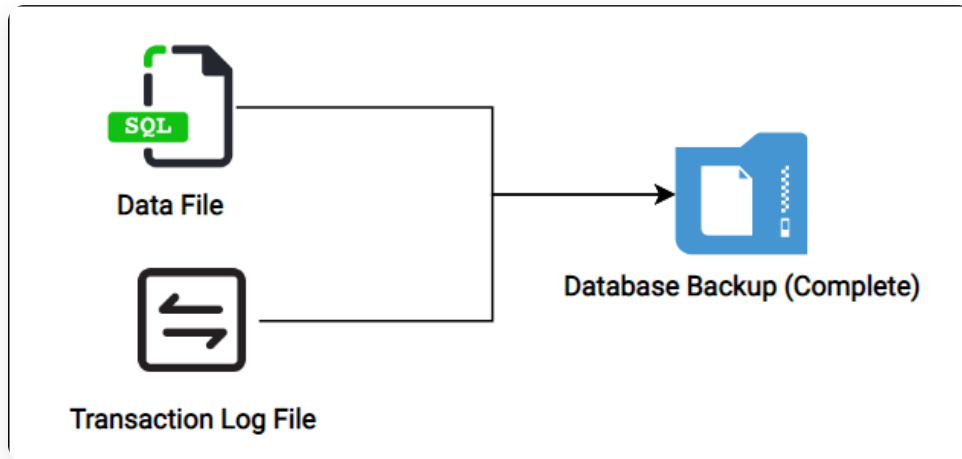
Faster restoration than incremental, less space than full

### ✗ Cons

More storage than incremental, slower restoration than full

**Best for:** *Medium-sized organizations*

# Recovery Models



## 🔄 Simple Recovery Model

No transaction log backups. Log space reused when checkpoint occurs.

### ✓ Pros

No administrative overhead

### ✗ Cons

Risk of data loss, no point-in-time restore

**Best for:** *Development/test databases, reporting databases*

---

## 🔄 Full Recovery Model

All transactions fully recorded in transaction log. Log sequence preserved.

### ✓ Pros

Minimal data loss, supports point-in-time restore

### ✗ Cons

Transaction log grows infinitely, requires regular log backups

**Best for:** *Mission-critical applications, high availability*

---

## 📑 Bulk-Logged Recovery Model

Similar to Full but bulk operations minimally logged.

### ✓ Pros

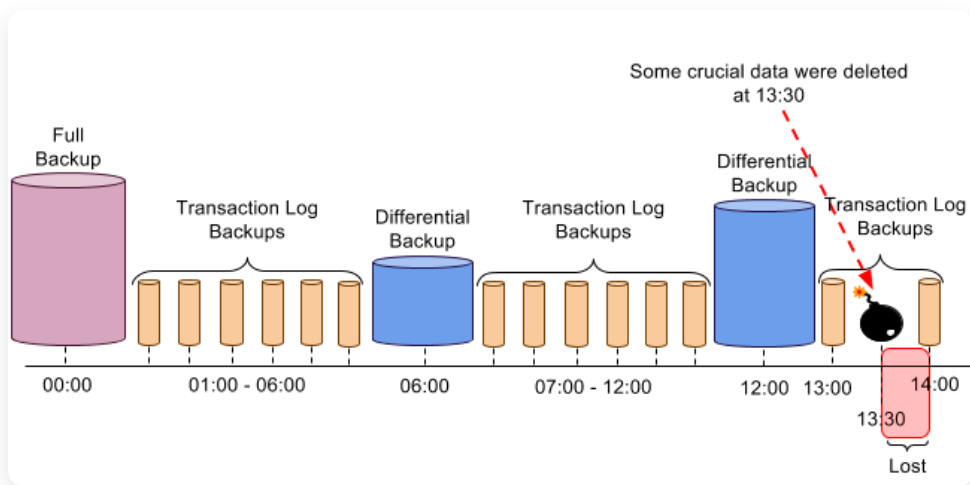
Better performance for bulk operations, reduces log space

### ✗ Cons

Cannot restore to specific point-in-time during bulk operations

**Best for:** *Databases with periodic bulk operations*

# Point-in-Time Recovery (PITR)



## ? What is PITR?

Allows restoring a database to a specific point in time, within seconds of precision

## ⚙️ How it Works

Creates a full backup first, then constantly backs up transaction logs. Recovery involves accessing the full backup and replaying transaction logs to the specified time.

## 👍 Benefits

- ✓ Minimizes data loss
- ✓ Supports precise recovery objectives
- ✓ Recovery to moments before failure

## ⚠️ Limitations

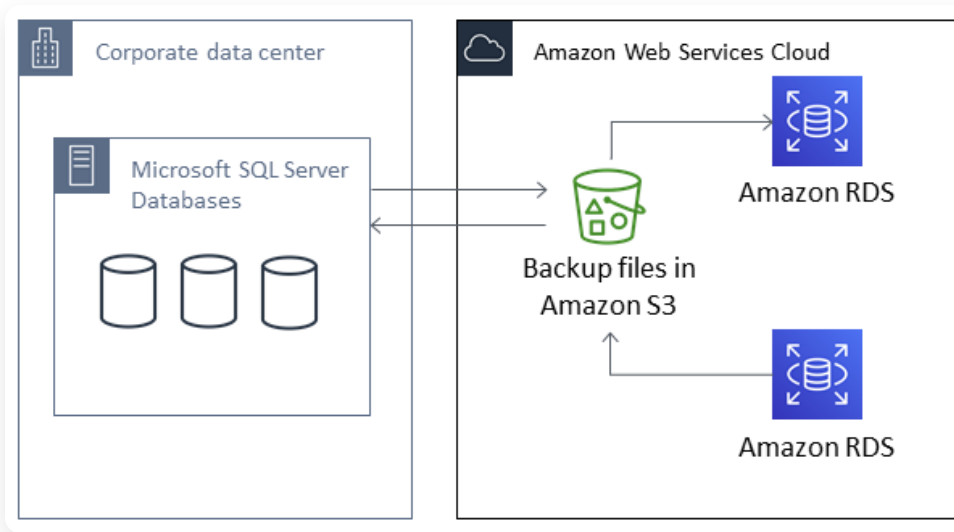
- 🕒 Max retention period (e.g., 35 days)
- ⚙️ Requires continuous backup config
- 🕒 Recent activity limitations

## 📊 Use Cases

- ★ Critical databases with minimal data loss tolerance
- 📈 Compliance requirements
- 👤 Recovering from human errors

# Backup Automation and Validation





## ! Importance of Automation

- 🕒 Reduces human error
- 🕒 Saves time
- 🔄 Ensures consistency
- ✅ Enables reliable testing

## ⚙️ How to Automate

- 💎 Use RESTful APIs
- 🔗 CI/CD integration
- 🔧 Orchestrate functionality
- 📜 PowerShell scripting

## ✅ Validation Techniques

- 🕒 Regular restore testing
- 🕒 RTO validation
- 🛡️ Backup integrity checks
- 🕒 RPO validation

## 💡 Best Practices

- 📅 Schedule regular automated backups
- 🔔 Implement monitoring and alerting
- 📄 Maintain documentation
- 🔄 Test recovery procedures regularly

# Summary and Best Practices

## 💡 Key Takeaways

- ✓ Choose backup types based on **data volume** and **recovery requirements**
- ✓ Select recovery models based on **data criticality** and **acceptable data loss**
- ✓ Implement PITR for databases requiring **minimal data loss**
- ✓ Automate backup processes to ensure **consistency** and **reliability**

## ★ Best Practices

- 📄 Follow the **3-2-1 rule**: 3 copies, 2 different media, 1 off-site
- 🕒 **Regularly test** backup restores to ensure recoverability
- 🔔 **Monitor** backup jobs and set up alerts for failures
- 📄 **Document** backup and recovery procedures
- 🔄 **Review and update** backup strategy as business needs change

## 👉 Recommendations

- ⬆️ **Combine different backup types** for optimal protection
- 🛡️ Use **full recovery model** for critical databases
- ⚙️ Implement **automated validation** of backup integrity
- 🕒 Establish clear **RTO** and **RPO** objectives