Planning and Implementing a Backup Strategy

Module Overview

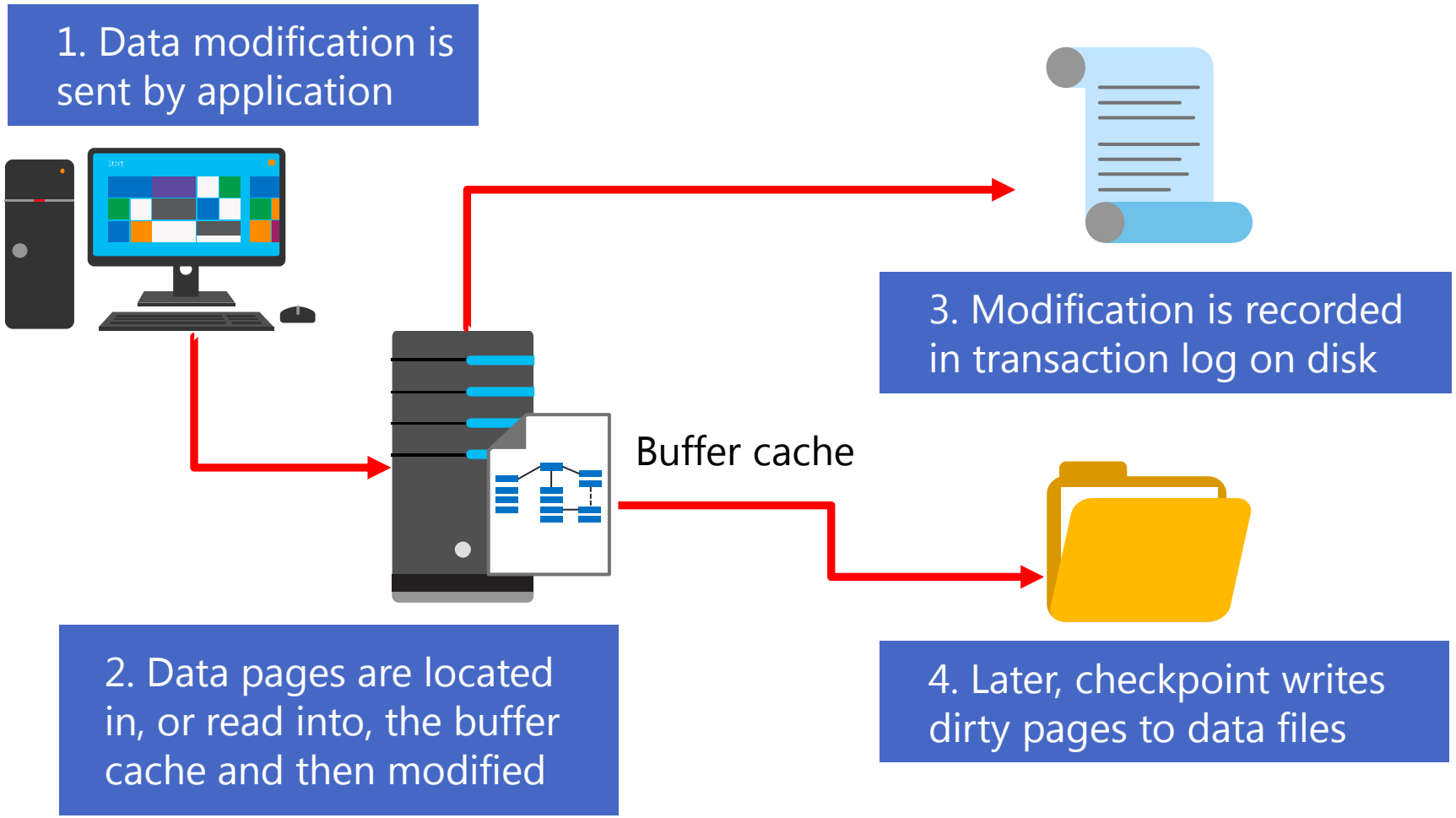
- Understanding SQL Server Recovery Models
- Planning a Backup Strategy
- Backing up Databases and Transaction Logs
- Using Backup Options
- Ensuring Backup Reliability

SQL Server Recovery Models

- Overview of SQL Server Transaction Logs
- Transaction Log File Structure
- Working with Recovery Models
- Capacity Planning for Transaction Logs
- Working with Checkpoint Options
- Demonstration: Logs and Full Recovery

Overview of SQL Server Transaction Logs

Transaction logs provide a history of actions executed by a database management system to guarantee atomicity and durability of transactions



Transaction Log File Structure

Sufficient information is logged to be able to:
- Roll back transactions if requested
- Recover the database in case of failure

Write Ahead Logging is used to create log entries
- Transaction logs are written in chronological order in a circular way
- Truncation policy for logs is based on the recovery model

Working with Recovery Models

Recovery Model	Description
Simple	• Does not permit or require log backups • Automatically truncates log to keep space requirements small
Full	• Requires log backups for manageability • Avoids data loss due to a damaged or missing data file • Permits recovery to a specified point in time
Bulk Logged	• Requires log backups for manageability • Can enhance the performance of bulk copy operations • Reduces log space usage by using minimal logging for many bulk operations

Capacity Planning for Transaction Logs

- Capacity needs are based on several factors:
  - Recovery model used for the database
  - Transaction log backup frequency in full and bulk logged recovery models
  - Number and size of transactions in the database
- Examine log behavior during pre-deployment testing

Working with Checkpoint Options

- Types of checkpoint operations:
  - Automatic
  - Indirect
  - Manual
  - Internal
- CHECKPOINT statement configures the target recovery duration

Planning a Backup Strategy

- Backup Types
- Determining Recovery Objectives
- Full Database Backup Strategies
- Transaction Log Backup Strategies
- Differential Backup Strategies
- Partial Backup Strategies
- Discussion: Planning Backup Strategies

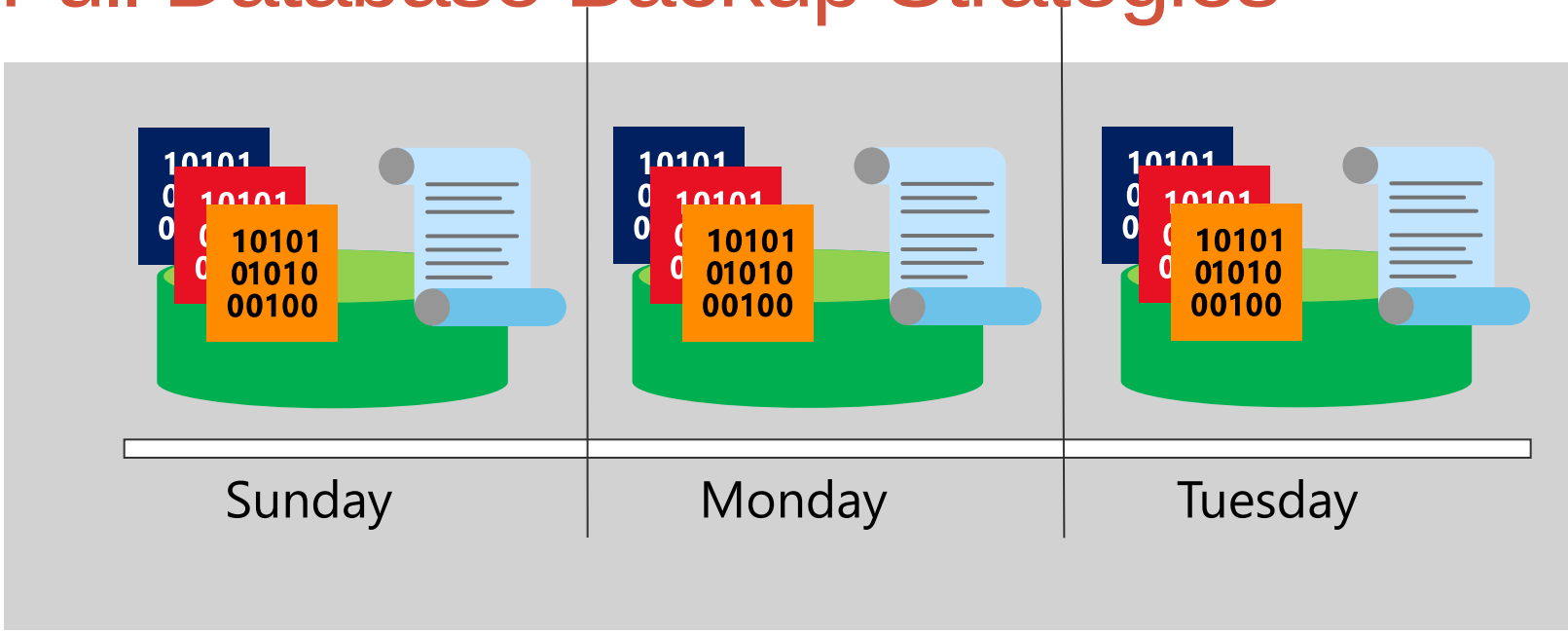
Backup Types

Backup type	Description
Full	All data files and the active part of the transaction log
Differential	The parts of the database that have changed since the last full database backup
Transaction Log	Any database changes recorded in the log files
File/File Group	Specified files or filegroups
Partial	The primary filegroup, every read/write filegroup, and any specified read-only filegroups
Tail-log Backup	Log backup taken of the tail of the log just before a restore operation
Copy Only	The database or log (without affecting the backup sequence)

Determining Recovery Objectives

- Determine safety levels:
  - How long can recovering take? (RTO)
  - How much data is it acceptable to lose? (RPO)
  - Is it possible to recover the data from other sources?
- Backup strategy should map to requirements:
  - Types and frequency of backups
  - Backup media to use
  - Retention period for backups and for media
  - Backup testing policy

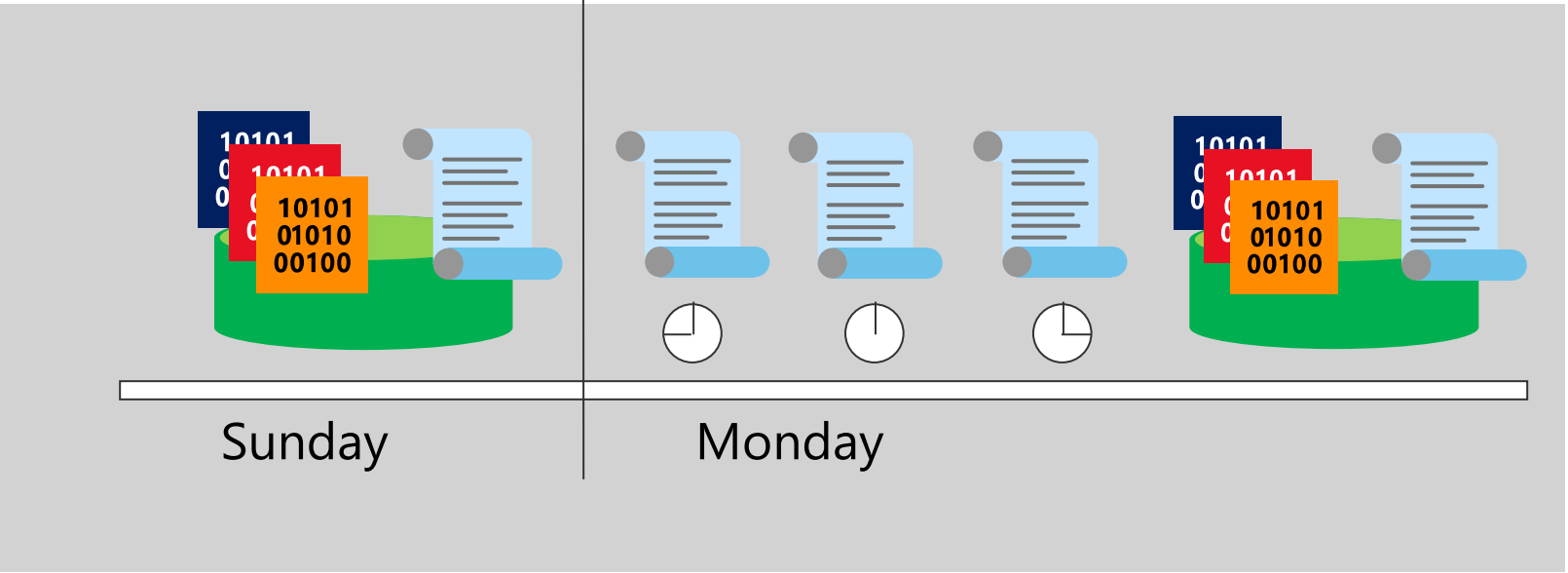
Full Database Backup Strategies



Full database backups:

- Back up all data and part of the log records
- Can be used to restore the whole database
- Permit recovery to backup times only

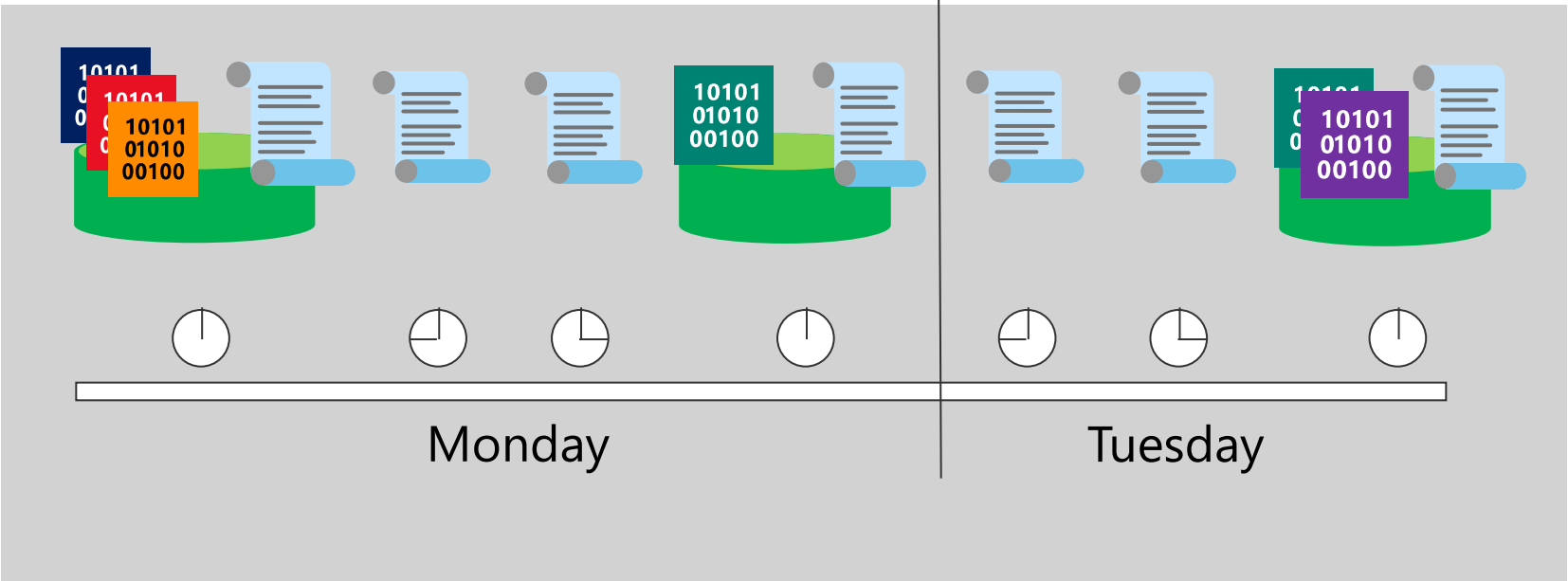
Transaction Log Backup Strategies



A database and transaction log backup strategy:

- Involves at least full and transaction log backups
- Enables point-in-time recovery
- Allows the database to be fully restored in the case of data file loss

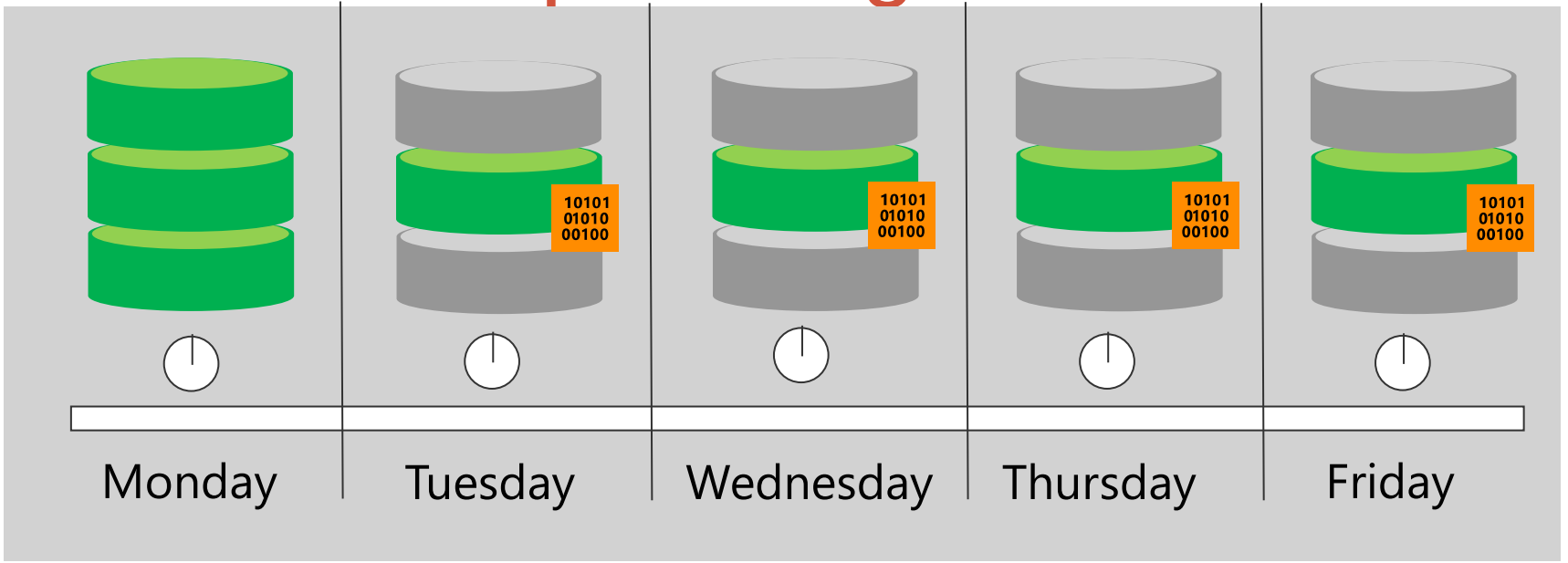
Differential Backup Strategies



A differential backup strategy:

- Involves performing full and differential database backups
- Includes differential backups with only changed data
- Is useful if only a subset of a database is modified more frequently than the rest of the database

Partial Backup Strategies



- Faster backup and restore for very large databases
- Can be complex to set up and manage

Discussion: Planning Backup Strategies

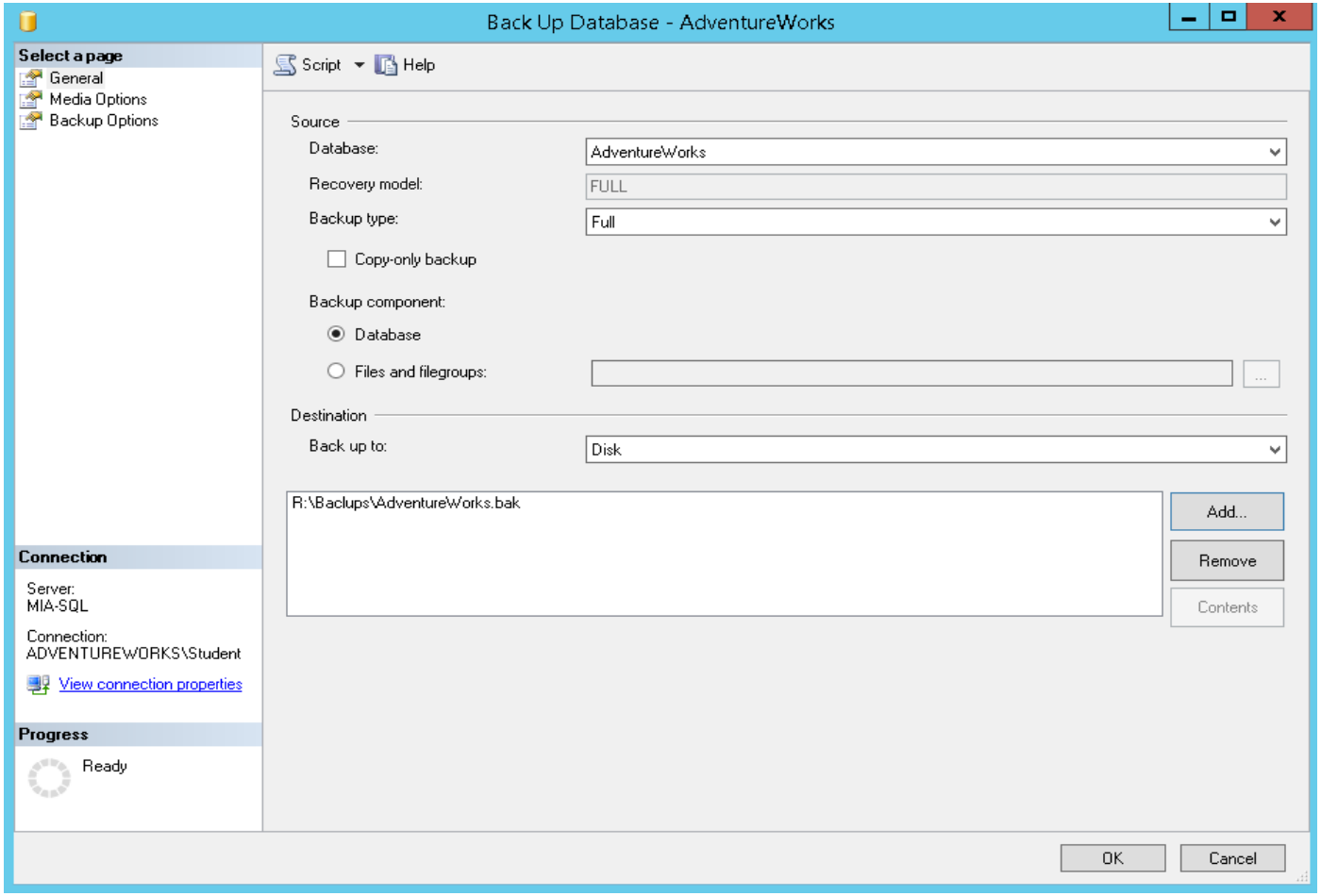
For each database in the scenario, determine:

- The appropriate recovery model
- The type(s) of backup to be performed
- The appropriate time(s) at which to perform the backup(s)

Backing up Databases and Transaction Logs

- Introduction to SQL Server Backup
- Media Sets and Backup Sets
- Performing Database Backups
- Performing Transaction Log Backups
- Performing Partial and Filegroup Backups
- Demonstration: Performing Backups

Introduction to SQL Server Backup




```
BACKUP { DATABASE | LOG] } <database_name>
TO <backup_device>, [,...n]
WITH <general_options>
```

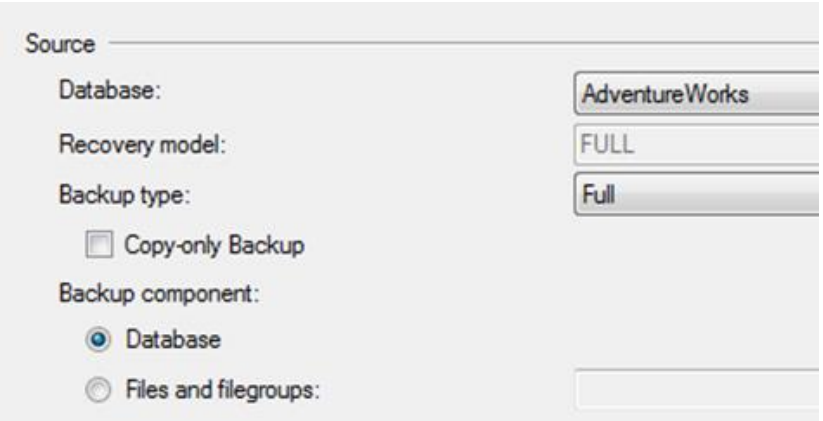
Performing Database Backups

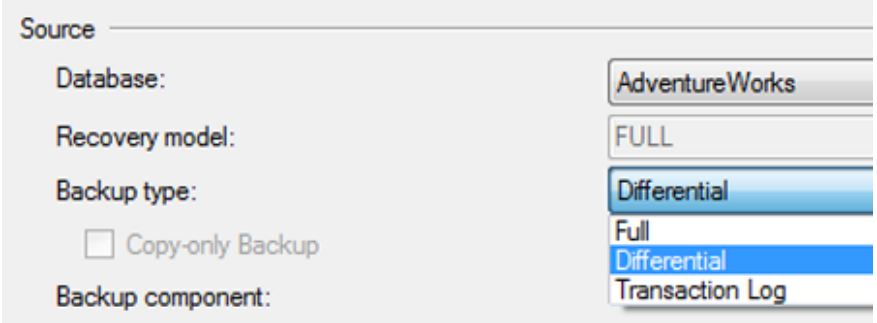
- Full Backup:
  - Entire database
  - Active portion of log file

Differential Backup

Extents modified since the last full database backup

Active portion of log file



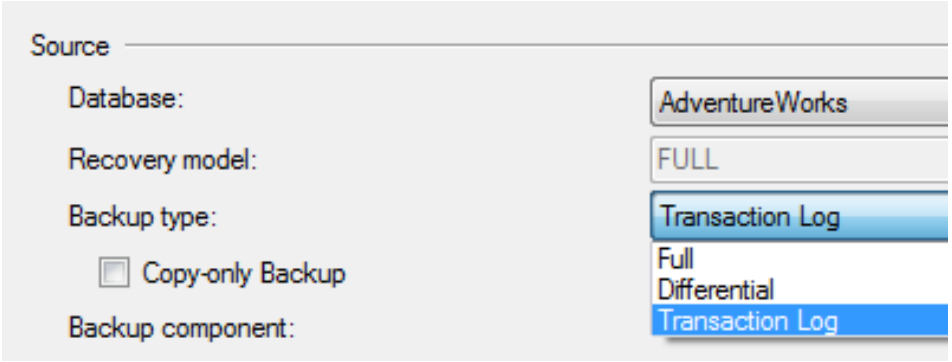



```
BACKUP DATABASE
AdventureWorks
TO DISK = 'R:\Backups\AW.bak'
WITH INIT;
```

```
BACKUP DATABASE
  AdventureWorks
TO DISK = 'R:\Backups\AW.bak'
WITH DIFFERENTIAL, NOINIT;
```

Transaction Log Backups

Backs up only the transaction log
Backs up the log from the last successfully executed log backup to the current end of the log
Truncates inactive log records unless options specified



Note: Database must be in full or bulk-logged recovery model

```
BACKUP LOG AdventureWorks
TO DISK = 'R:\Backups\AW.bak'
WITH NOINIT;
```

```
BACKUP LOG AdventureWorks
TO DISK = 'R:\Backups\AW.bak'
WITH [NORECOVERY | NO_TRUNCATE | CONTINUE_ON_ERROR];
```

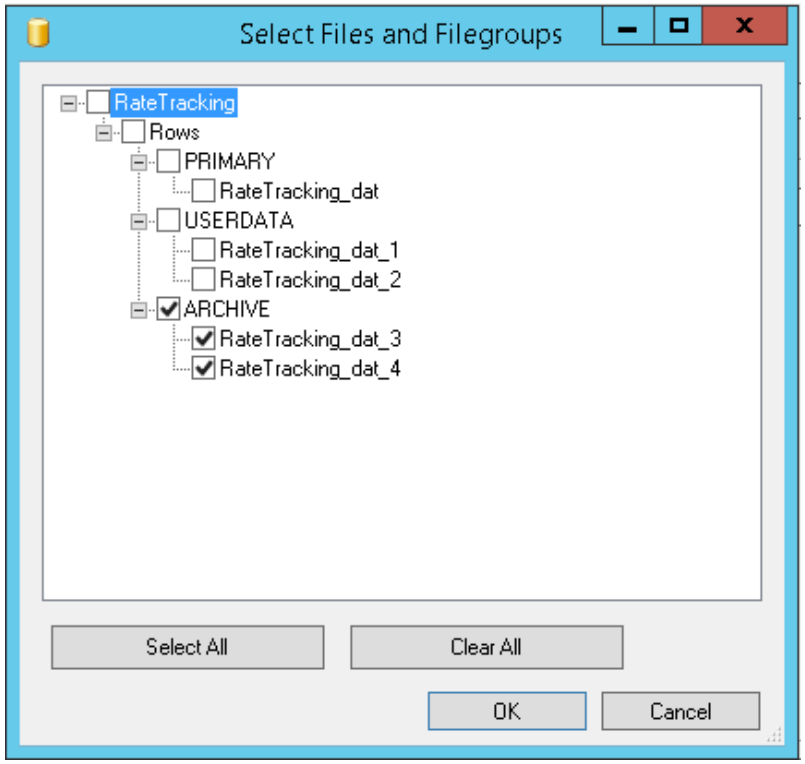
Partial and Filegroup Backups

- Partial Backup
  - Primary filegroup
  - Read/Write filegroups

```
BACKUP DATABASE LargeDB
READ_WRITE_FILEGROUPS
TO DISK = 'R:\Backups\LrgRW.bak'
WITH INIT;
```

- File or Filegroup backup
  - Specific files or filegroups

```
BACKUP DATABASE LargeDB
FILEGROUP = 'FG2'
TO DISK = 'R:\Backups\LrgFG2.bak'
```



Demonstration: Performing Backups

In this demonstration, you will see how to:

- Perform a full database backup
- Perform a differential database backup
- Perform a transaction log backup

Lesson 4: Using Backup Options

- Copy-Only Backups
- Compressing Backups
- Demonstration: Using Backup Compression
- Encrypting Backups
- Demonstration: Using Backup Encryption

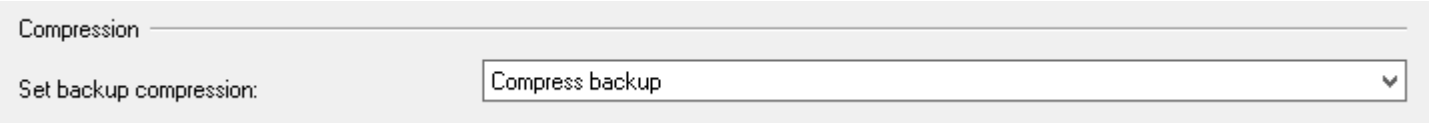
Copy-Only Backups

Back up the database without changing the restore order
Copy-only transaction log backups do not truncate the log
Copy-only full database backups do not affect the differential base


```
BACKUP DATABASE AdventureWorks
TO DISK = 'Q:\Backups\AW_Copy.bak'
WITH COPY_ONLY, INIT;
```

Compressing Backups

- Reduces size of backup on device
- Reduces I/O requirements, increases CPU usage
- Increases speed of backup and restore
- Some restrictions:
  - Cannot share media with uncompressed backups
  - Cannot share media with Windows backups
  - Cannot be restored to pre-2008 SQL Server versions




```
BACKUP DATABASE AdventureWorks
TO DISK = 'R:\Backups\AW_Comp.bak'
WITH COMPRESSION;
```

Demonstration: Using Backup Compression

In this demonstration, you will see how to:

- Use backup compression.

Lesson 5: Ensuring Backup Reliability

- Options for Ensuring Backup Integrity
- Viewing Backup History
- Retrieving Backup Metadata
- Demonstration: Verifying Backups

Ensuring Backup Integrity

CHECKSUM backup option

- Available for all backup types
- Generates a checksum over the backup stream
- Use to verify the backup

Backup verification

- Can use RESTORE VERIFYONLY for backup verification
- Useful when combined with the CHECKSUM option

Viewing Backup History

- SQL Server tracks all backup activity in a set of tables in the **msdb** database

```
SELECT      bs.media_set_id, bs.backup_finish_date,  bs.type,
            bs.backup_size, bs.compressed_backup_size,
            mf.physical_device_name
FROM dbo.backupset AS bs
INNER JOIN dbo.backupmediafamily AS mf
ON bs.media_set_id = mf.media_set_id
WHERE database_name = 'AdventureWorks'
ORDER BY backup_finish_date DESC;
```

- The **Backup and Restore Events** report in SQL Server Management Studio displays detailed backup history information

Retrieving Backup Metadata

- RESTORE LABELONLY returns information about the backup media on a specified backup device
- RESTORE HEADERONLY returns all the backup header information for all backup sets on a particular backup device
- RESTORE FILELISTONLY returns a list of data and log files contained in a backup set

Demonstration: Verifying Backups

In this demonstration, you will see how to:

- View the backup and Restore Events report.
- Query backup history tables.
- Verify backup media.