# Ensuring Database Durability: Core Mechanisms

A Learning Guide

October 26, 2025

**Abstract**

This document outlines the core techniques used by database systems to ensure durability. Durability is the guarantee that once a transaction is committed, its data will survive system crashes, power failures, or unexpected restarts. We will explore Write-Ahead Logging (WAL), Shadow Paging, Checkpointing, and Storage Synchronization.

## Introduction

Databases ensure durability—the guarantee that committed data survives crashes, power failures, and unexpected system restarts—through several key mechanisms that work together to protect your data.

## 1 Write-Ahead Logging (WAL)

**Write-Ahead Logging** is the most fundamental and widely-used technique for ensuring durability. WAL follows a simple but powerful principle: all modifications are first written to an append-only log file on disk before they are applied to the actual database pages. This sequential log contains both redo information (to replay committed transactions) and undo information (to rollback incomplete transactions).

When a database crashes mid-operation, the recovery process compares the WAL against the database state. Since the WAL is always ahead of the actual database, the system can replay all committed transactions from the log to restore a consistent state. This technique provides several benefits: it requires fewer disk writes since only log records need immediate flushing, sequential I/O for logging is much faster than random disk updates, and it enables point-in-time recovery by replaying historical transactions.

## 2 Shadow Paging

**Shadow Paging** is an alternative copy-on-write technique that maintains two versions of database pages simultaneously. The shadow page table points to the original, unmodified database pages representing the consistent pre-transaction state, while the current page table tracks new copies of modified pages. When a transaction modifies data, changes are written to new page copies rather than updating pages in-place.

If the transaction commits successfully, the current page table becomes the new shadow, making changes durable atomically. If the transaction fails or the system crashes, the database simply discards the modified pages and reverts to the shadow page table—no undo or redo operations needed. While conceptually simpler than WAL, shadow paging can lead to storage fragmentation and is less commonly used in modern systems.

## 3 Checkpointing

**Checkpointing** is a critical optimization technique that works alongside logging mechanisms to speed up recovery. Without checkpoints, recovering from a crash would require scanning the entire log file from the beginning, which could take an extremely long time.

During a checkpoint operation, the database forces all log records from memory to stable storage, flushes all modified database pages to disk, and writes a special checkpoint record in the log. When recovery is needed after a crash, the system only needs to scan the log from the last checkpoint forward, since all transactions committed before that point are already permanently saved. This dramatically reduces recovery time while maintaining durability guarantees.

## 4 Storage Synchronization

Beyond these core techniques, databases use additional mechanisms to ensure data truly reaches persistent storage. The **fsync** system call forces data from operating system buffers to physical disk. Checksums verify data integrity after writes. Storage redundancy through replication provides additional protection against hardware failures. Together, these mechanisms ensure that once a transaction is committed, its changes are permanent and will survive any subsequent system failure.