
# TECHNICAL REPORT WRITING

## *KINDS OF FAILURES, FAILURE CONTROL METHODS*

**NAME:** RUPAK SARKAR

**MAKAUT ROLL NO.:** 14271024036

**STREAM:** MASTER OF COMPUTER APPLICATION

**SUBJECT NAME:** RELATIONAL DATABASE MANAGEMENT SYSTEM

**SUBJECT CODE:** MCAN-102

**COLLEGE NAME:** MEGHNAD SAHA INSTITUTE OF TECHNOLOGY

**UNIVERSITY NAME:** MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY

# SUMMARY

In this report, we will learn about the different types of failures in database system and also study about methods that are used to control the failures in database systems.

# CONTENTS



# INTRODUCTION

**What are Database Failures?**

Database failures refer to situations where the normal functioning of a database is disrupted, leading to issues like data loss, corruption, or inaccessibility. These failures can occur due to various reasons, including hardware malfunctions, software bugs, user errors, or external factors such as cyberattacks or natural disasters. They impact the reliability, availability, and consistency of a database.

**Key Characteristics of Database Failures**

1. **Impact on Data Integrity:**
    - Data might become inconsistent, corrupted, or lost.
2. **Interruption of Services:**
    - Applications relying on the database may face downtime or malfunction.
3. **Recovery Required:**
    - Affected databases often need to be repaired, restored from backups, or rebuilt.

# KINDS OF FAILURES IN DATABASE SYSTEM

The different types of database failures are as follows:

**1. Transaction Failures**

- **Reasons:**
  - Errors during transaction processing, such as logical errors in the application.
  - Violations of constraints, like primary key or foreign key constraints.
  - Deadlocks that force a rollback.
- **Impact:** The current transaction is aborted, but the database remains consistent.

**2. System Failures**

- **Reasons:**
  - Operating system crashes.
  - Hardware failures (CPU, memory, etc.).
  - Software bugs in the database management system (DBMS).
- **Impact:** The database server shuts down unexpectedly, requiring recovery upon restart.

**3. Media Failures**

- **Reasons:**
  - Disk crashes or corruption.
  - File system failures.
  - Loss of storage devices containing database files.
- **Impact:** Loss of data stored on the affected media. Recovery often involves restoring backups.

**4. Concurrency Failures**

- **Reasons:**
  - Improper management of concurrent transactions.
  - Race conditions leading to data inconsistencies.
- **Impact:** Data anomalies like lost updates, dirty reads, or uncommitted dependencies.

**5. Logical Failures**

- **Reasons:**
  - Human errors, such as accidental data deletion or incorrect updates.
  - Bugs in application programs that interact with the database.
- **Impact:** Incorrect data or schema modifications.

**6. Configuration Failures**

- **Reasons:**
  - Misconfigured database settings.
  - Errors during migration or updates.
- **Impact:** Suboptimal performance or system crashes.

**7. Network Failures**

- **Reasons:**
  - Connectivity issues between the database and client applications.
  - Network partitioning in distributed databases.
- **Impact:** Interrupted transactions, delays, or inconsistency in replicated databases.

**8. Security Breaches**

- **Reasons:**
  - Unauthorized access or hacking.
  - Insider threats or data leaks.
  - Malware or ransomware attacks.
- **Impact:** Data loss, corruption, or theft.

**9. Power Failures**

- **Reasons:**
  - Sudden power outages without proper backup systems.
- **Impact:** Abrupt shutdown of the database server, requiring restart and recovery.

**10. Natural Disasters**

- **Reasons:**
  - Events like earthquakes, floods, or fires.
- **Impact:** Physical damage to data centers, resulting in significant data loss if backups are not geographically redundant.

# FAILURE CONTROL METHODS IN DATABASE SYSTEM

Failure control methods in databases are mechanisms designed to ensure data consistency, durability, and reliability even in the event of transaction failures, system crashes, or other unexpected events. These methods are part of the recovery management system in a Database Management System (DBMS). Below are the primary failure control methods:

**1. Transaction Management**

Transactions are managed using **ACID properties** to maintain database integrity:

- **Atomicity**: Ensures a transaction is either fully completed or not executed at all.
- **Consistency**: Ensures the database transitions from one valid state to another.
- **Isolation**: Prevents intermediate transaction states from being visible to other transactions.
- **Durability**: Guarantees that once a transaction is committed, its changes persist even in the event of failure.

**2. Logging**

Logging involves recording all changes made by transactions in a separate log file to support recovery:

- **Write-Ahead Logging (WAL)**: Ensures changes are logged before being applied to the database.
- **Types of Logs**:
    - **Undo Logs**: Records data before modification to enable rollback.
    - **Redo Logs**: Records data after modification to reapply changes during recovery.

**3. Checkpoints**

Checkpoints are periodic snapshots of the database state:

- Reduces recovery time by creating a point of reference for the recovery process.
- During recovery, only transactions after the last checkpoint need to be processed.

**4. Shadow Paging**

- A **shadow copy** of the database is maintained during a transaction.
- Changes are applied to a separate copy (shadow page table), and the original remains unchanged until the transaction is committed.
- Advantages: No logs are needed.
- Disadvantages: High overhead for copying.

**5. Backup and Recovery**

- **Regular Backups**: Periodically saving the entire database to restore it in case of catastrophic failure.
- **Incremental Backups**: Backing up only the changes made since the last backup to reduce storage overhead.
- Recovery involves restoring the backup and applying logs to reach the last consistent state.

**6. Concurrency Control Mechanisms**

Concurrency control prevents conflicts in multi-user environments:

- **Locking Protocols**: Controls access to data using locks (e.g., two-phase locking).
- **Timestamp Ordering**: Ensures transactions are executed in timestamp order to avoid conflicts.
- **Deadlock Detection and Resolution**: Identifies and resolves deadlocks using timeout or wait-for graphs.

**7. Fail-Safe Measures**

Fail-safe techniques ensure minimal disruption:

- **Deferred Update**: Changes are applied to the database only after the transaction commits.
- **Immediate Update**: Changes are applied immediately but are logged to allow rollback if needed.

**8. Redundancy Mechanisms**

- **Mirroring**: Copies of the database are maintained on multiple servers.
- **Replication**: Data is duplicated across multiple locations to ensure availability.

---

# CONCLUSION

Database systems are essential for reliable and consistent data management, even in the face of failures. Failures such as system crashes, transaction errors, and disk or network issues pose significant challenges, requiring robust control methods. Techniques like transaction management, logging, checkpointing, shadow paging, backups, and concurrency control ensure data integrity and uphold the ACID properties.

By implementing these methods, databases maintain reliability, security, and availability, enabling organizations to operate efficiently. A solid understanding of failure types and recovery strategies is crucial for designing resilient systems capable of meeting the demands of modern data management.

# BIBLIOGRAPHY

https://en.wikipedia.org/

# ACKNOWLEDGEMENT

I would like to acknowledge all those without whom this project would not have been successful. Firstly, I would wish to thank our subject teacher Mrs. Aparna Datta who guided me throughout the project and gave her immense support. She made us understand how to successfully complete this project and without her, the project would not have been complete.

This project has been a source to learn and bring our theoretical knowledge to the real-life world. So, I would really acknowledge her help and guidance for this project.

I would also like to thank my parents who have always been there whenever needed. Once again, thanks to everyone for making this project successful.