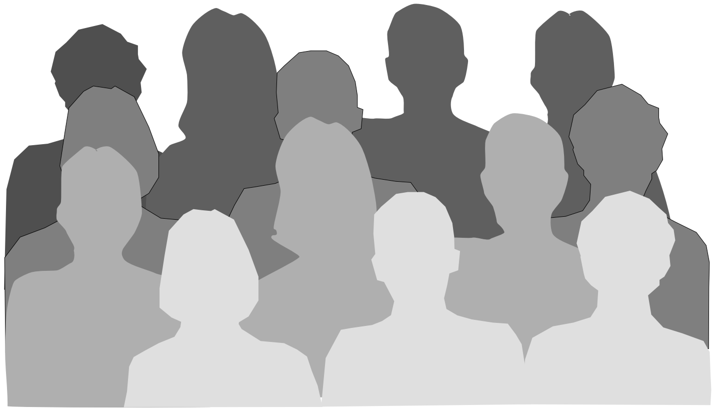
IBM DB2/2: Forward Recovery & User Exit

IBM Field Television Network
May 19, 1993



Presented by:

Jeffrey W. Fisher

Product Planner

Guylaine G. Cantin

Workstation Database Developer

IBM Toronto Development Lab

Overview

- Concepts
- Forward Recovery
- User Exit
- Operational Considerations
- Tuning
- Demo

Concepts

What are the Database Logs?

- **A configurable number of files that record:**
  - changes to the data and index files during normal processing

insert/delete/update row
add/delete/update index key
create/drop table

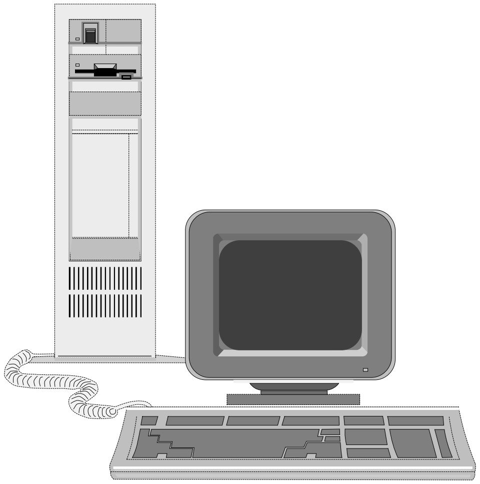
  - changes in the transaction state

Commit
Rollback

When/Why Are Logs Used?

- The database logs are used to return a databas to a "consistent" state should a failure occur
- A database is marked as "inconsistent" when the first log record is written for the database. The database remains "inconsistent" until the final STOP USING on the database

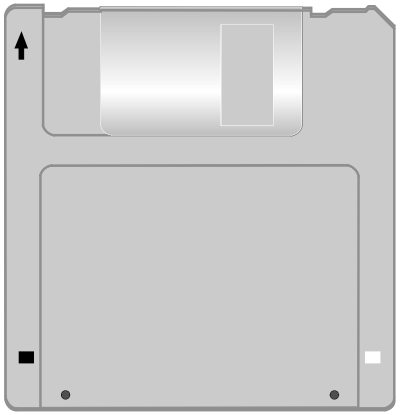
Types of Failure



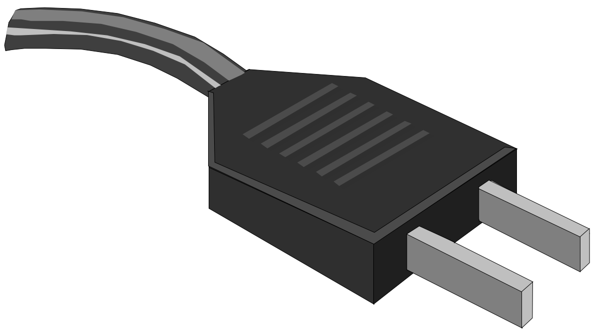
HARDWARE

MEDIA

OPERATIONAL



SOFTWARE



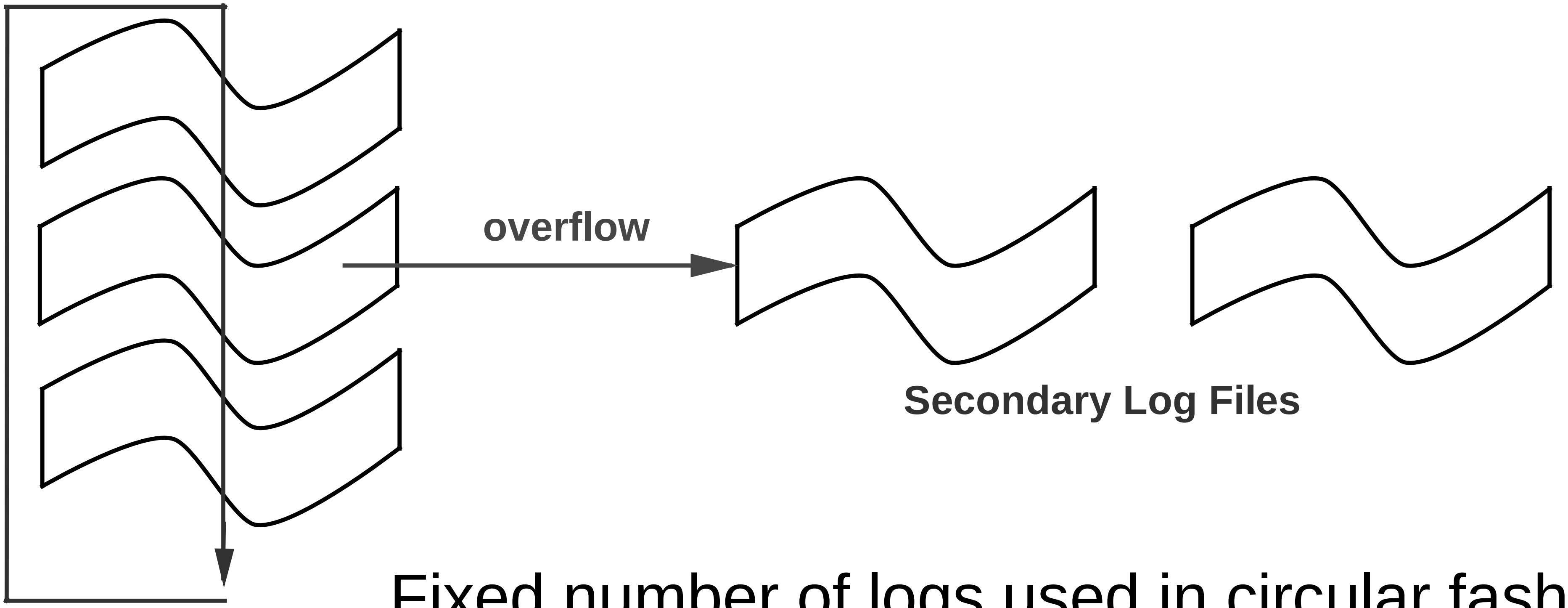
POWER

IBM *DB2/2*™

Log File Naming

- Name format is Snnnnnnn.LOG
- Sequence starts at 0000000, wraps at 9999999
- Lowest active log file tracked in LOGHEAD
- Next log to be used in NEXTACTIVE
- All held in LOGPATH directory

Circular Logs



Fixed number of logs used in circular fashion

LOGPRIMARY preallocated

LOGSECOND allocated as needed (overflow

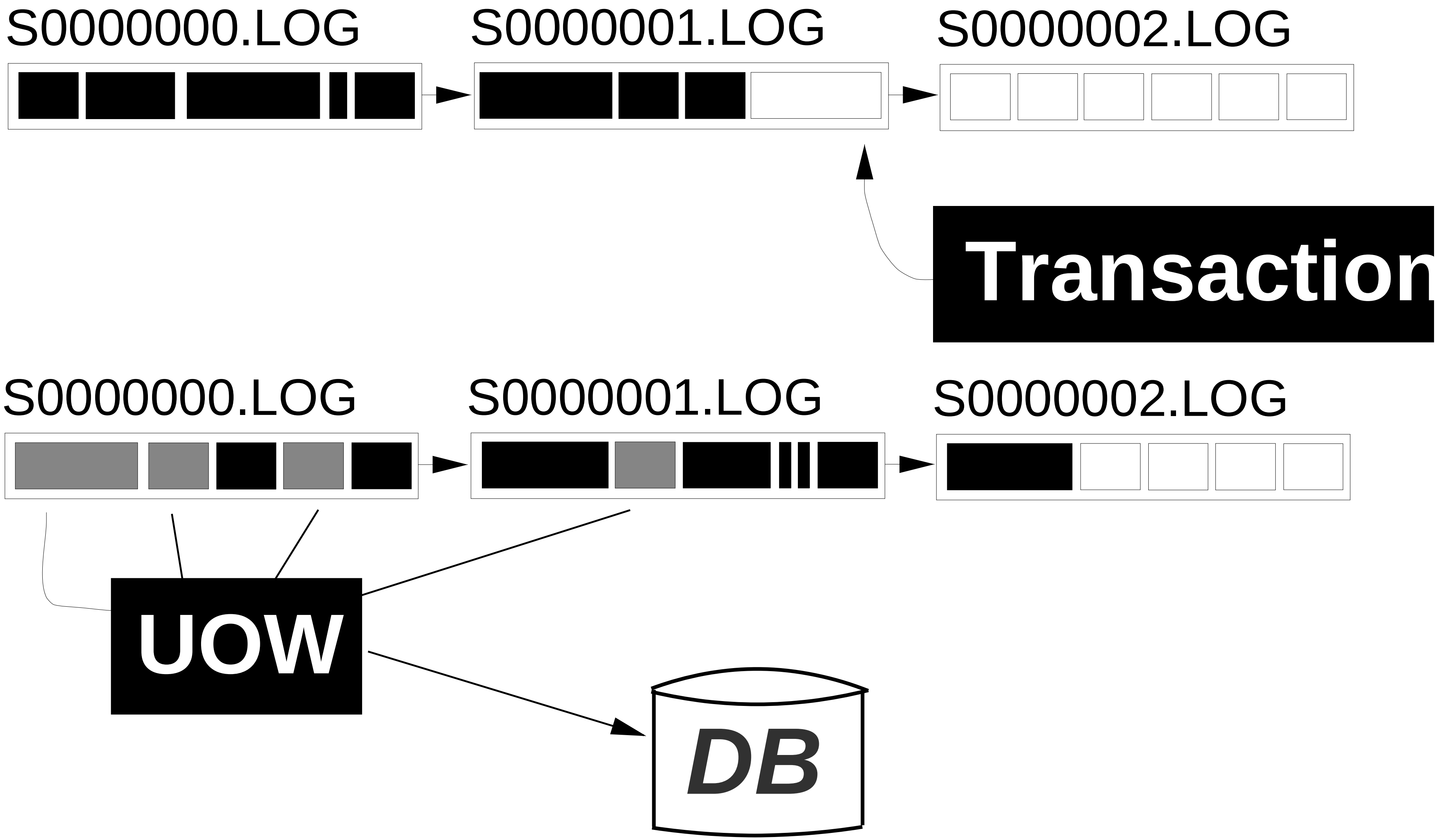
Size set by LOGFILSZ

Space freed when data committed on disk

Freed space reused

Not suitable for Forward Recovery

Circular Logging



S0000000.LOG

S0000001.LOG

S0000002.LOG

Transaction

S0000000.LOG

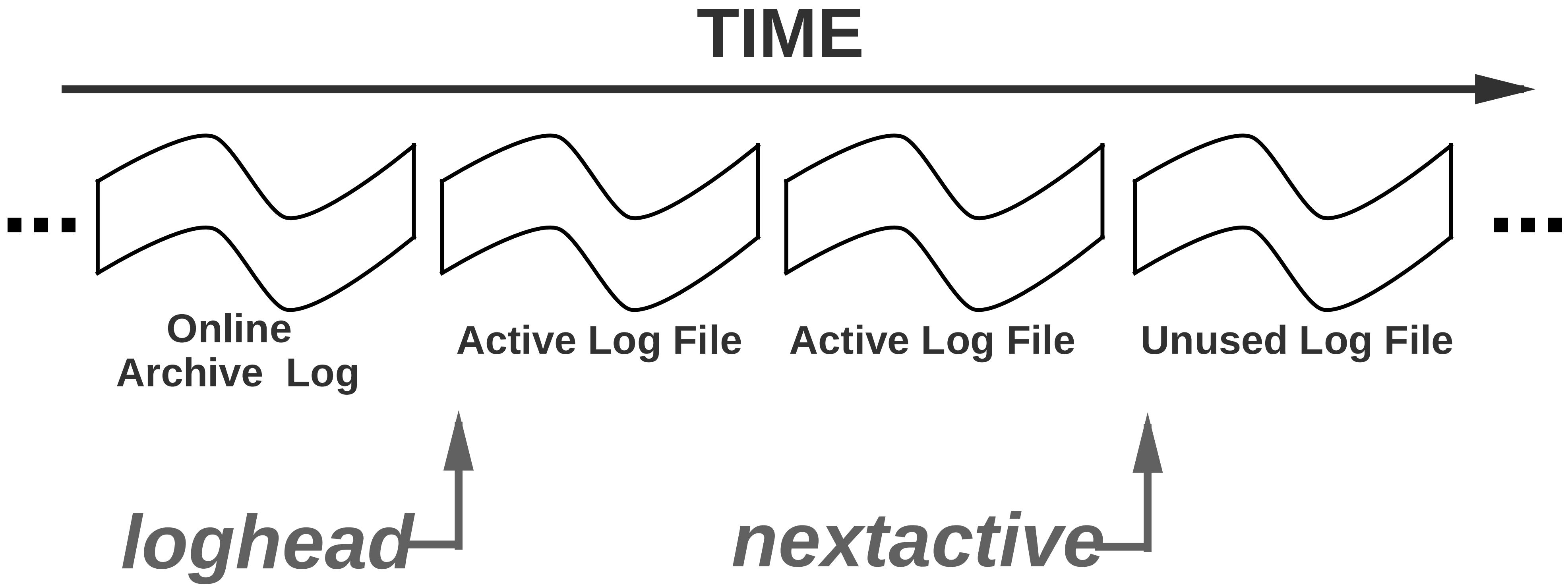
S0000001.LOG

S0000002.LOG

UOW

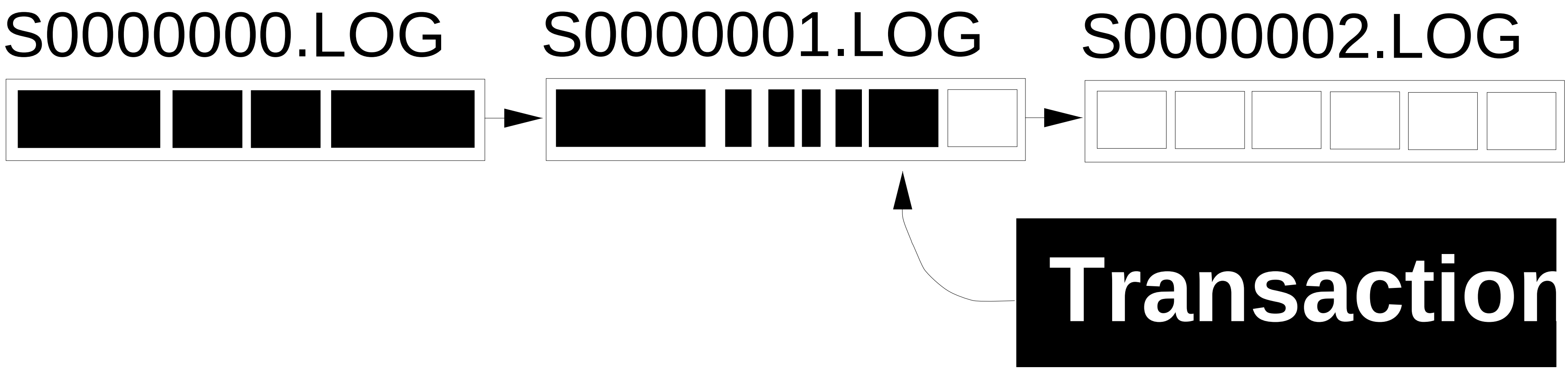
DB

Retained Logs



Enable Roll Forward Recovery
Log files are not discarded
As logs fill new ones are allocated

Archival Logging



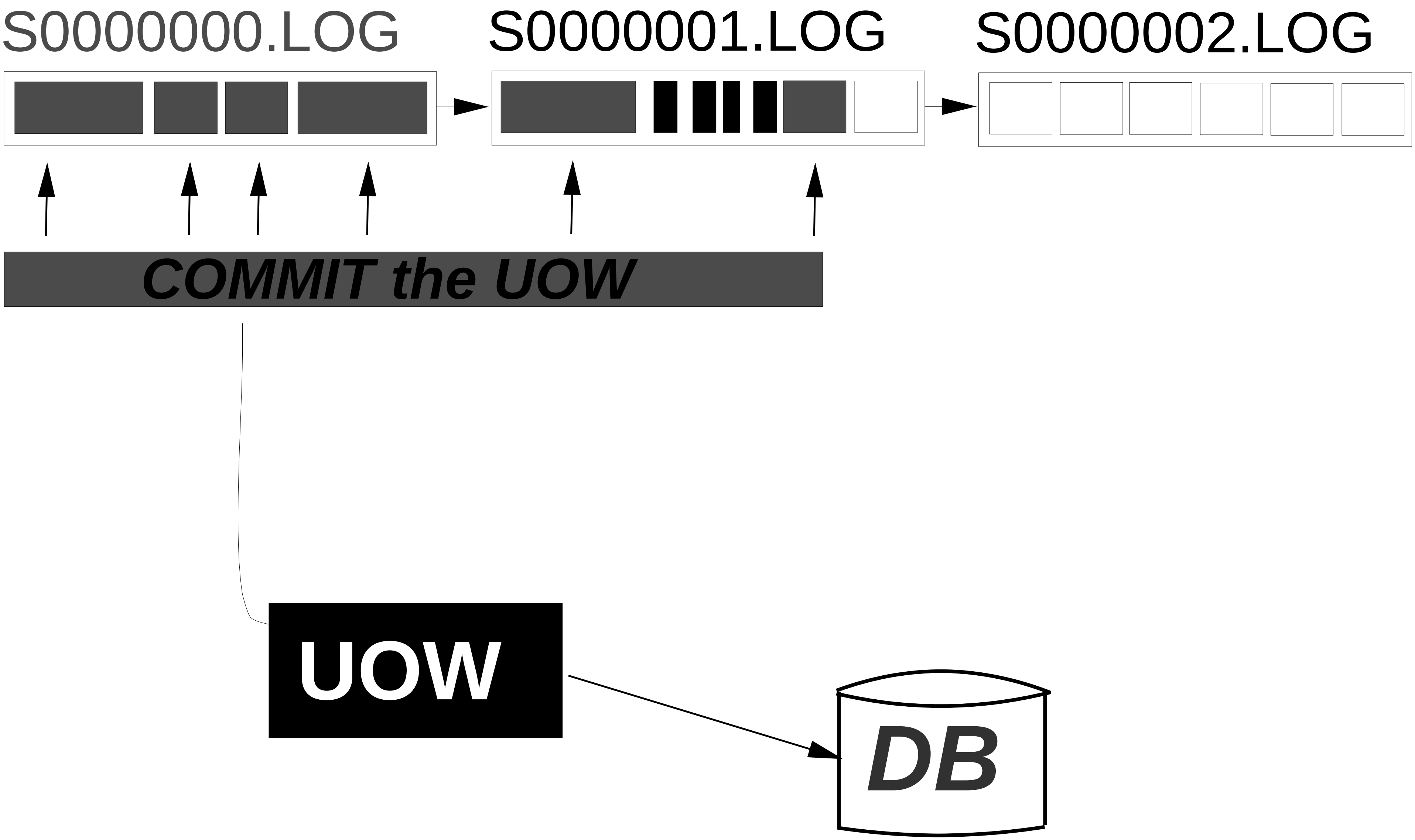
S0000000.LOG

S0000001.LOG

S0000002.LOG

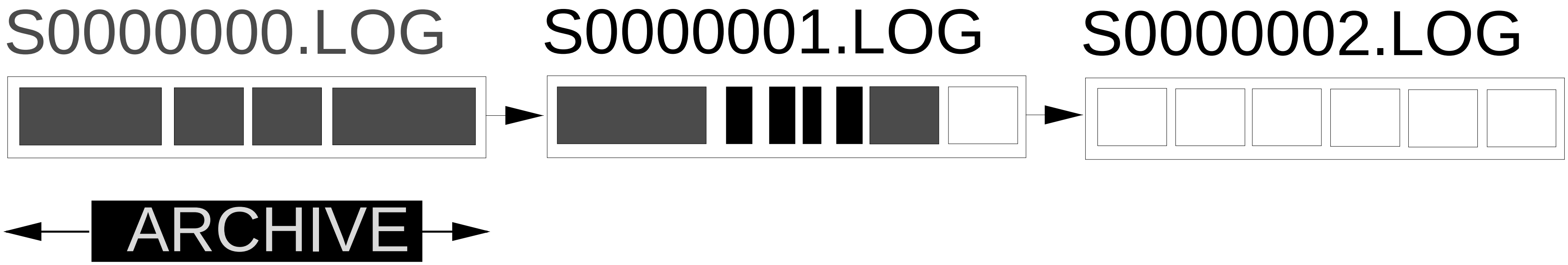
Transaction

Archival Logging....



IBM DB2/2™

Archival Logging....



S0000000.LOG

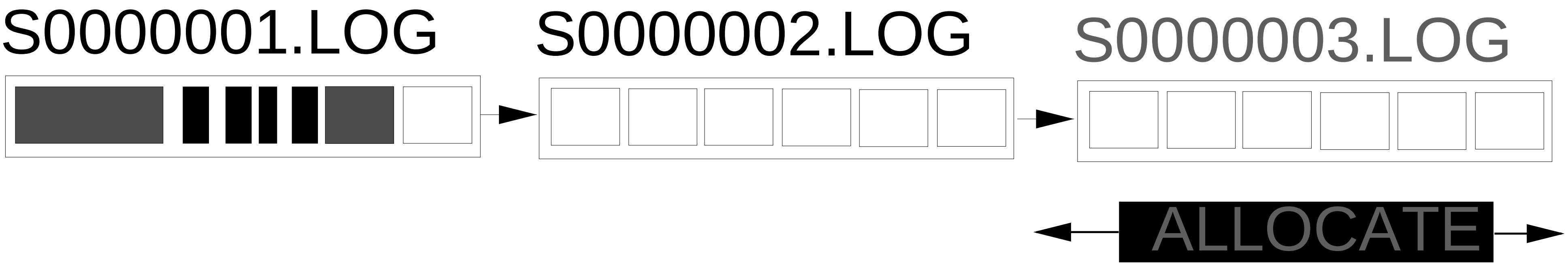
S0000001.LOG

S0000002.LOG

ARCHIVE

IBM DB2/2™

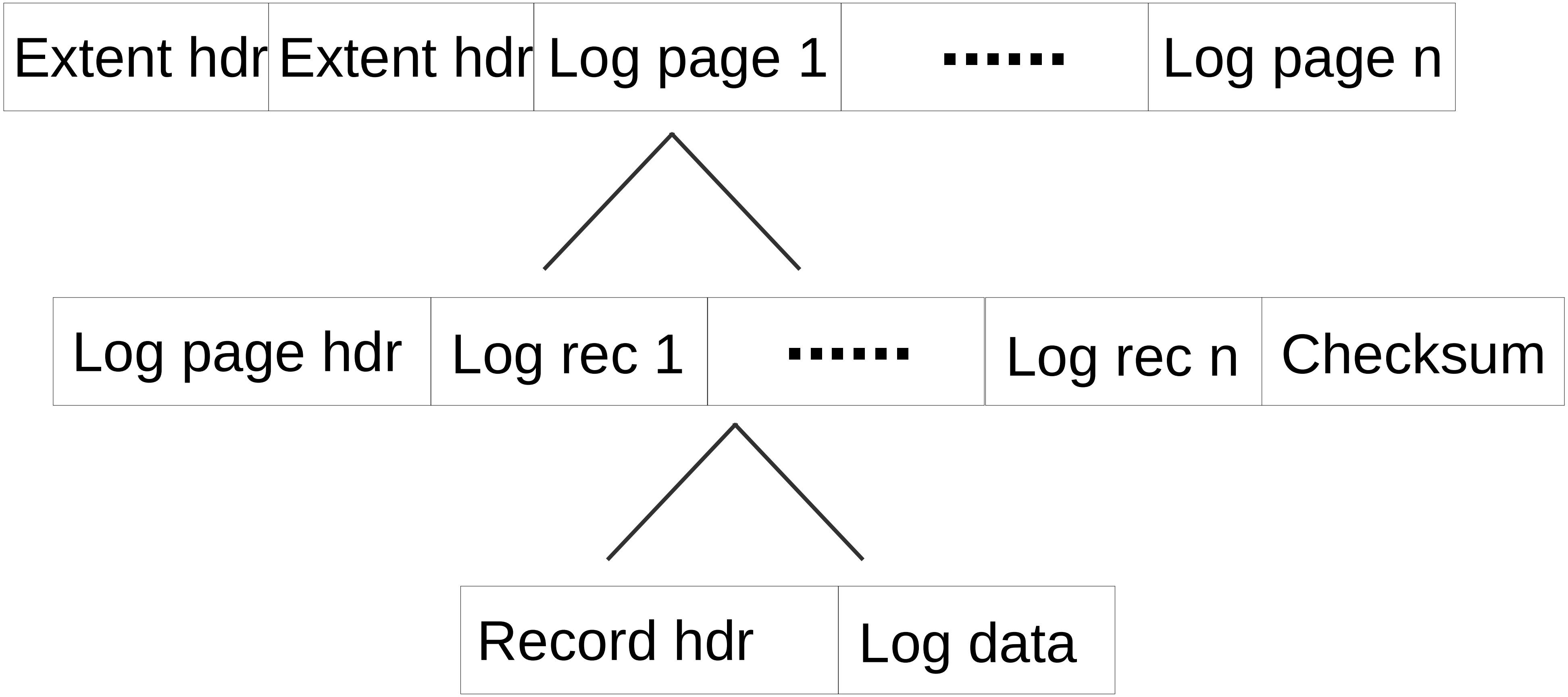
Archival Logging....



Log File Structure

S0000015.LOG

size = LOGFILSZ



Extent hdr | Extent hdr | Log page 1 | ······ | Log page n

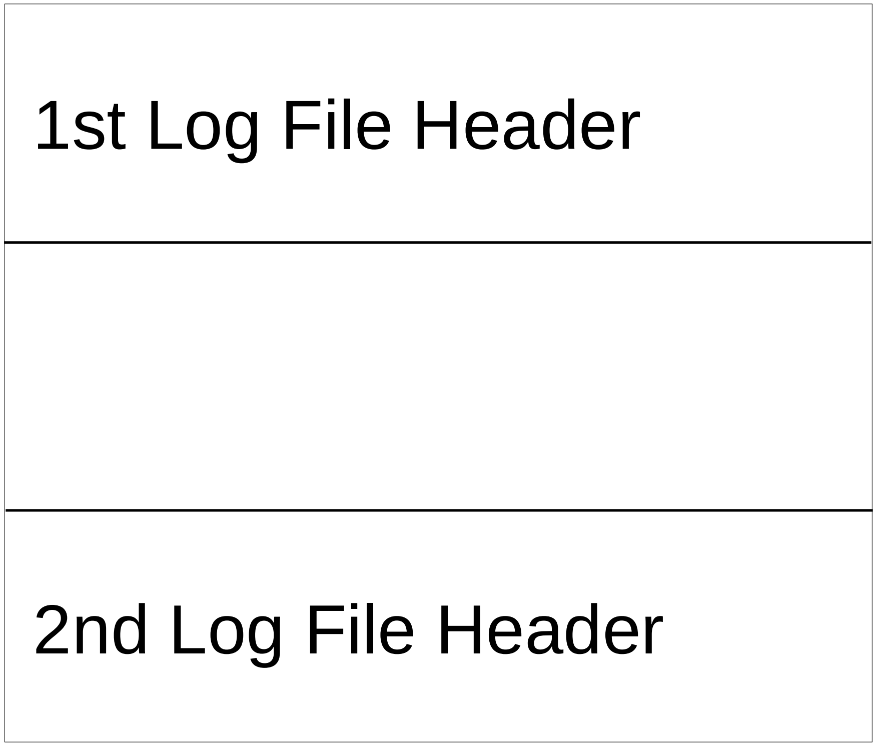
Log page hdr | Log rec 1 | ······ | Log rec n | Checksum

Record hdr | Log data

DB2/2™

Log File Header

SQLOGCTL.LFH size = 2968



1st Log File Header

2nd Log File Header

Contains Log Control Info

- Log Configuration
- Log files to be archived
- Active Logs
- Start point for crash recovery

Kept in database directory

Log Record Types

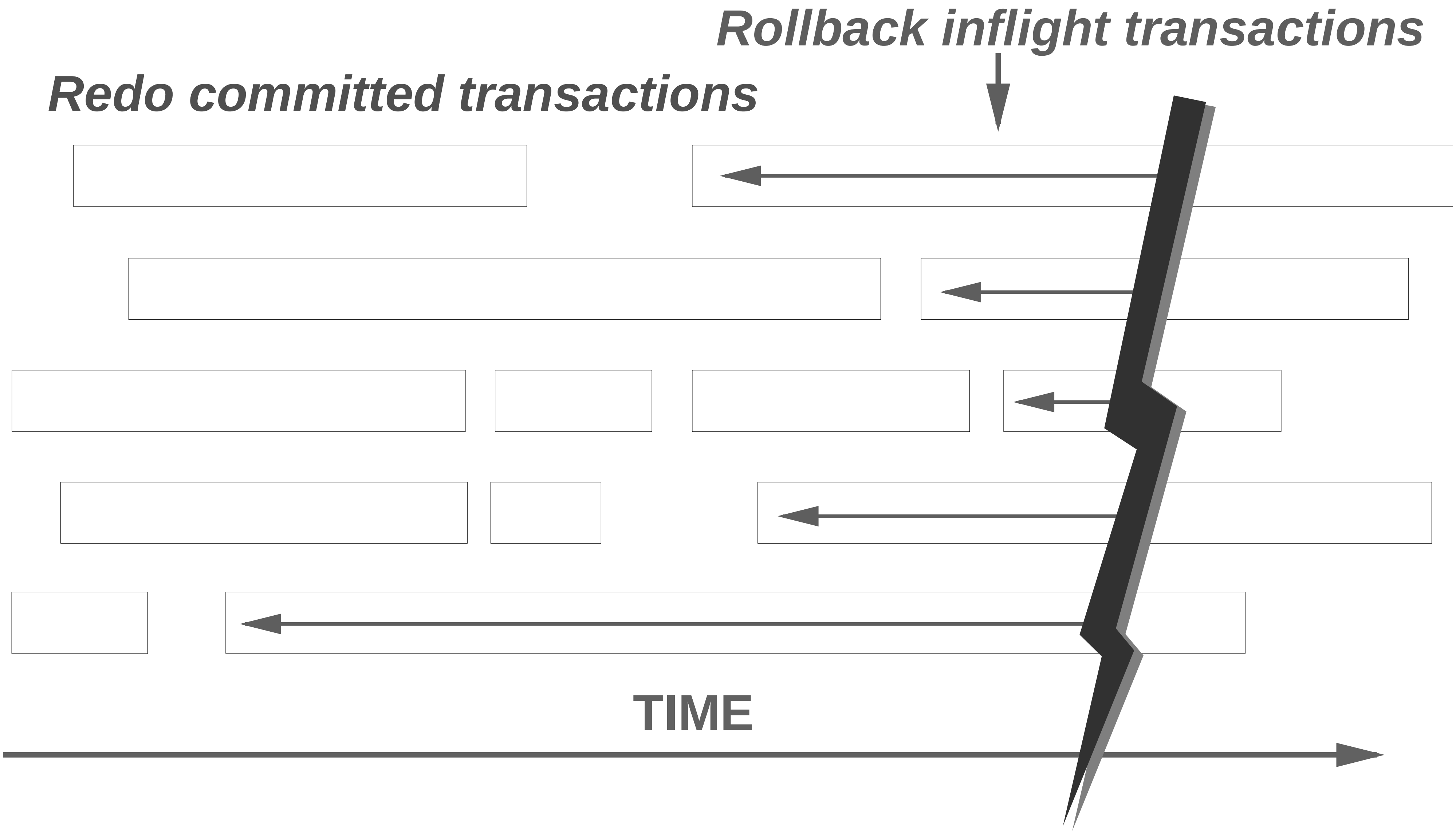
- **UNDO**
  - info used to undo a trans should a ROLLBACK occur
- **REDO**
  - contains the "new" info to be placed in the database
- **NORMAL**
  - contains both UNDO and REDO information
- **COMMIT**
  - indicates DB is in consistent state
- **ABORT**
  - marks successful completion of a ROLLBACK

IBM DB2/2™

3 Recovery Options

- Crash Recovery
- Version Recovery
- Roll Forward Recovery

Define: Crash Recovery



Rollback inflight transactions

Redo committed transactions

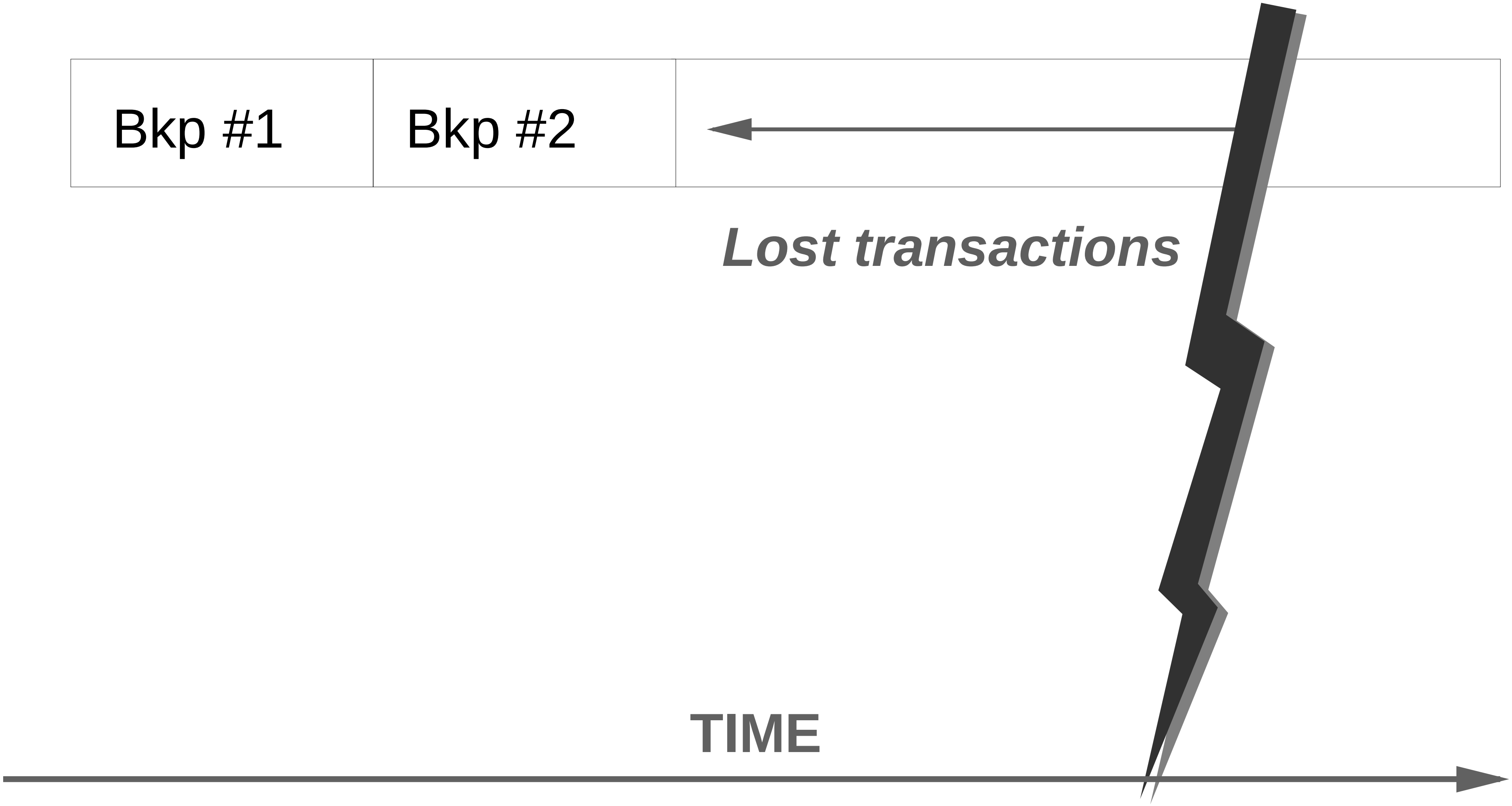
TIME

DB2/2™

Crash Recovery

- **Required after abnormal termination**
  - – i.e. no explicit STOP USING on the DB
- **will require a RESTART**
  - – use DBM RESTART API
  - – Automatic with AUTORESTART enabled
- **Process: RESTART uses active logs to**
  - – (1) redo all committed transactions
  - – (2) undo all inflight transactions

Define: Version Recovery



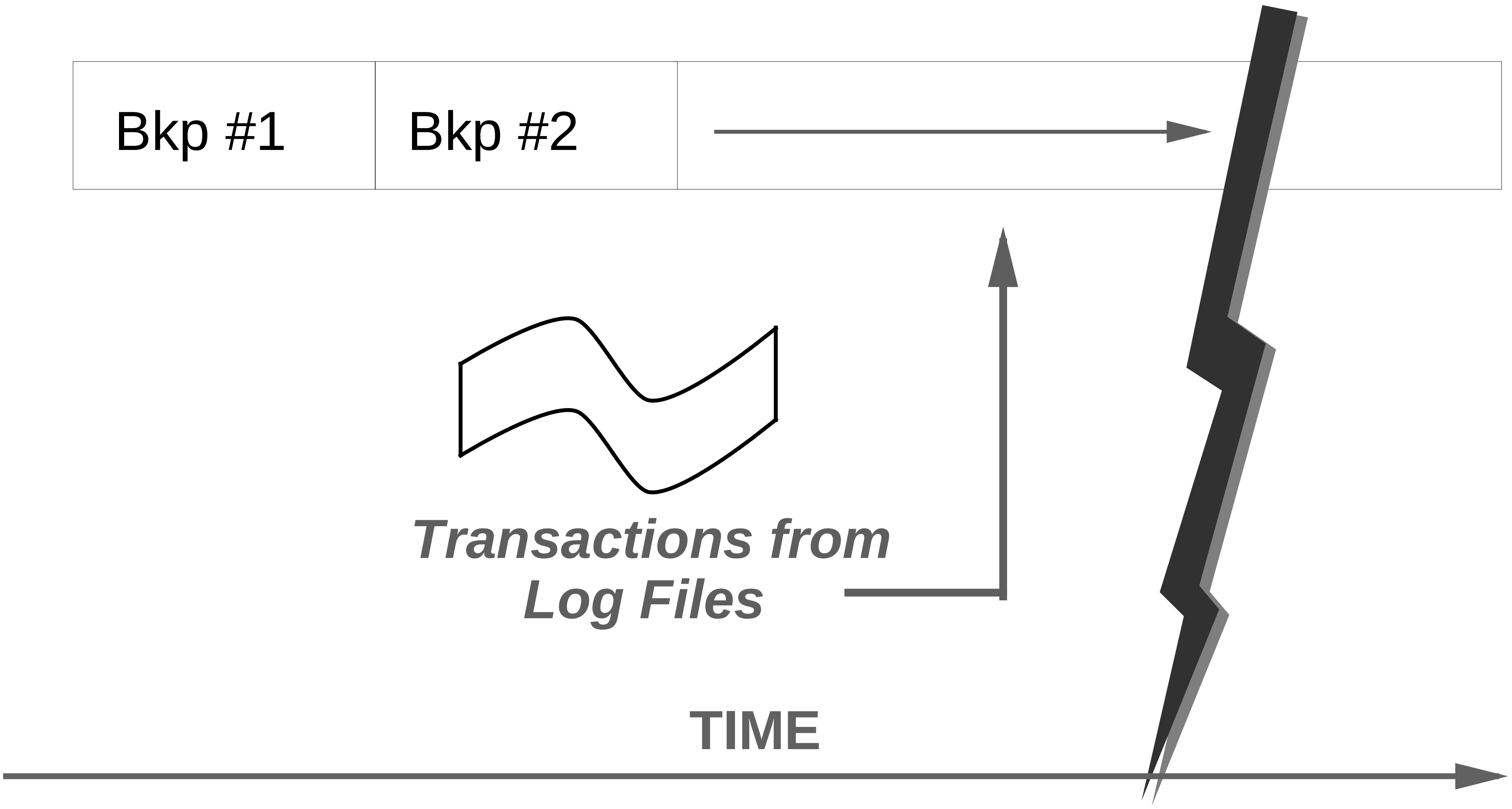
IBM DB2/2™

Version Recovery

- **Required after total loss of the database**
- **Recovers DB to point when backup was taken**
  - requires frequent backups to be effective
- **Changes to DB since backup are lost**
- **Process:**
  - (1) RESTORE database

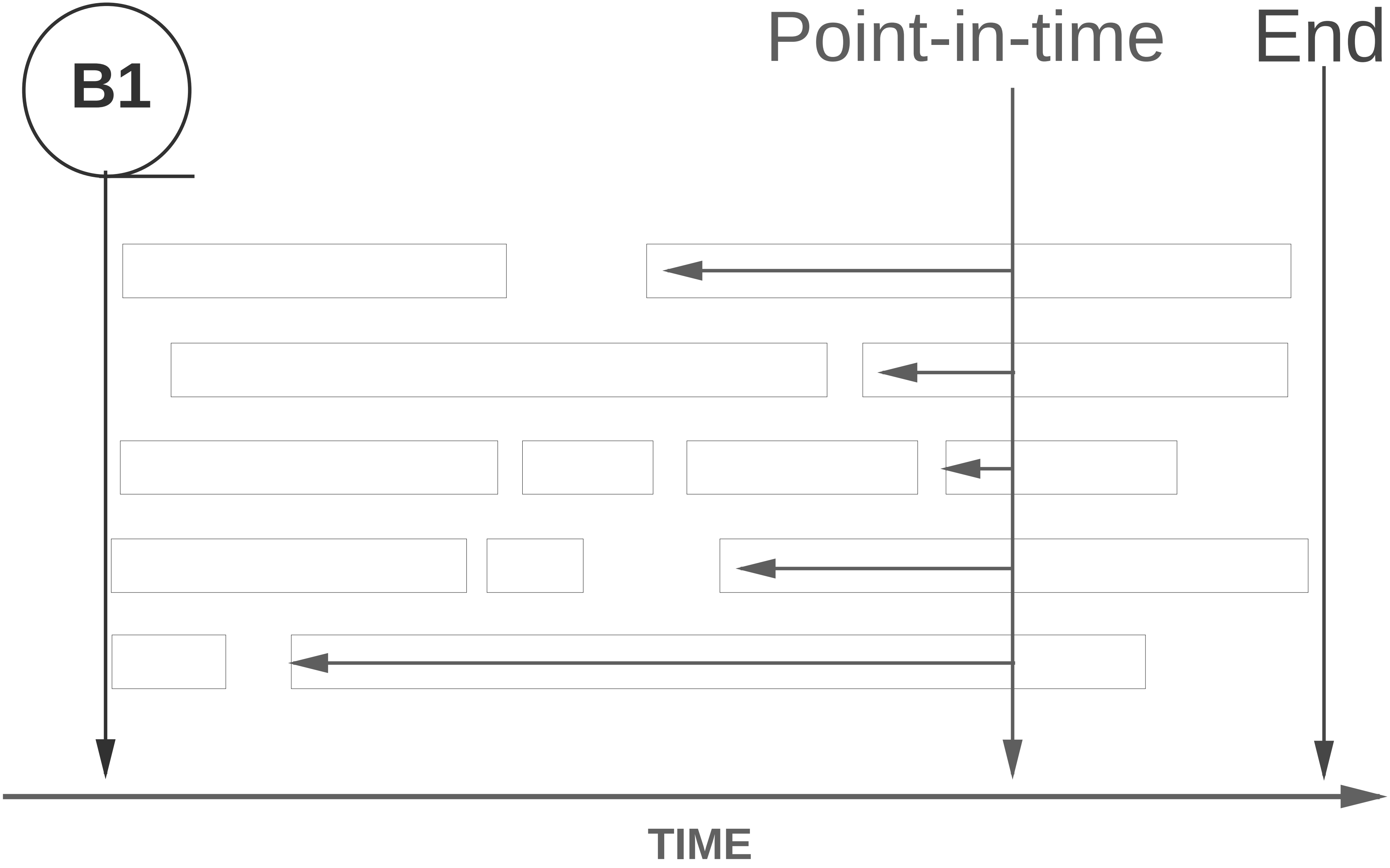
Forward Recovery

Define: Roll Forward Recovery



IBM DB2/2™

Roll Forward Options



Roll Forward Recovery

- Required after a serious error situation occurs or users need to restore the database back to a specific point in time
- Benefit: allows recovery of all data possible up to the point of the failure
- Rolls through transaction information containe in the archived logs
- Enabled by setting LOG_RETAIN and/or USER_EXIT flag in DB Configuration

Roll Forward Process

- **Restore database from prior backup**
- **User initiated recovery to point-in-time or end o logs**
- **Retrieve and roll through logs in sequential order to reapply data:**
  - (1) redo all committed transactions
  - (2) undo all inflight transactions
- **Recovery stops at specified point or at missing log file**

User Exit

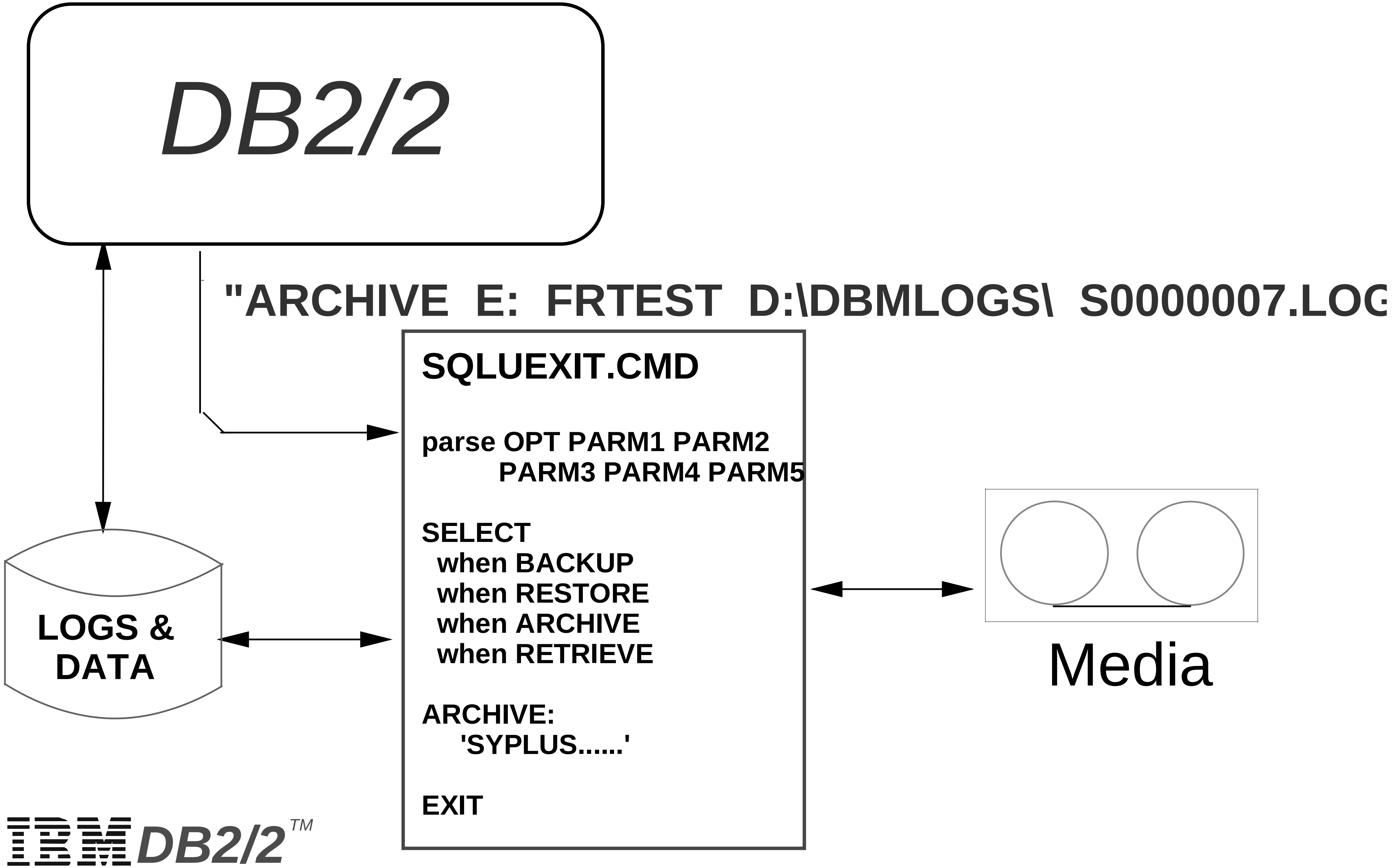
User Exit

- Benefit: allows for the usage of devices not supported by OS/2
- Provides great flexibility
- Support both Backup & Restore and Log Archi & Retrieve
- Implemented as a called program "SQLUEXIT"
- One User Exit used for entire database system

User Exit

- **4 Rexx Samples Supplied in /SQLLIB:**
  - XCOPY
  - SYTOS
  - Mountain Corp.'s FILESAFE
  - Maynard Corp.'s MAYNSTREAM
- **May require customization by DBA**
  - site standards
  - operational reporting
  - multiple databases
  - Non-standard device specific requirements

User Exit



DB2/2

"ARCHIVE E: FRTEST D:\DBMLOGS\ S0000007.LOG

SQLUEXIT.CMD

```
parse OPT PARM1 PARM2
        PARM3 PARM4 PARM5

SELECT
  when BACKUP
  when RESTORE
  when ARCHIVE
  when RETRIEVE

ARCHIVE:
    'SYPLUS......'

EXIT
```

LOGS & DATA

Media

IBM DB2/2™

User Exit Output

0 Successful

4 Unknown Problem
8 Operator intervention required

Requeue for retry

12 Hardware error
16 Defect/Configuration/Install error
20 Parameter error
28 Unknown error
32 Operator cancelled

Send Alert

Parms: ARCHIVE

- **OPT:** *ARCHIVE*
- **Parm1:** *D:*
- **Parm2:** *FRTEST*
- **Parm3:** *D:\DBMLOGS\*
- **Parm4:** *S0000000.LOG*
- **Parm5:** *(null)*

"ARCHIVE D: FRTEST D:\DBMLOGS\ S0000007.LOG"

Archiving Log Files

- **Circumstances for archival:**
  - log file is full (even if active)
  - last STOP USING on the database
- **User Exit performs the archive**
  - Copy log file from LOGPATH directory to offline archiv location or device
- **Enabled by setting USER_EXIT flag in DB conf**
- **Log file is deleted by DBM from the LOGPATH when no longer active**
- **Log file is truncated before archiving**

Backup: Basic vs. Advanced

- Basic
  - Dependent on and limited by the OS/2 Backup API
  - OS/2 Supported Devices only (Diskette, Partition, LAN Drive)
  - Limited flexibility
  - Best suited for SMALL backups
  - Initiated by DBM Backup API to "Drive X:"

- Advanced
  - Dependent on a user-written exit "SQLUEXIT" program
  - Any non-standard destinations or devices that can be coded into the User Exit
  - Very flexible
  - Suitable for ANY backup of any size
  - Initiated by DBM Backup API to "Drive 0"

Advanced BACKUP: 2 Phases

- **Phase 1**
  - UIF File: physical location of the DB
  - Example: "SQL00001.UIF FRTEST D:\SQL00001\"
- **Phase 2**
  - The contents of SQLxxxxx directory
    - SQLDBCON
    - SQL00001.SEM
    - SQL00001.DAT
    - SQL00002.DAT
    - SQL00002.LF
    - SQLOGCTL.LFH

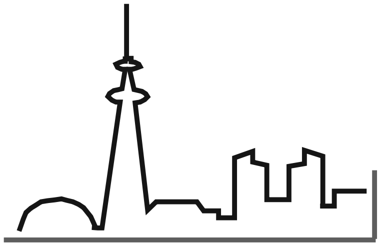
Parms: BACKUP Phase 1

- **OPT:** *BACKUP*
- **Parm1:** *D:*
- **Parm2:** *FRTEST*
- **Parm3:** *D:\SQLDBDIR\SQL00001.BKP*
- **Parm4:** *FRTEST-701479476*
- **Parm5:** *1*

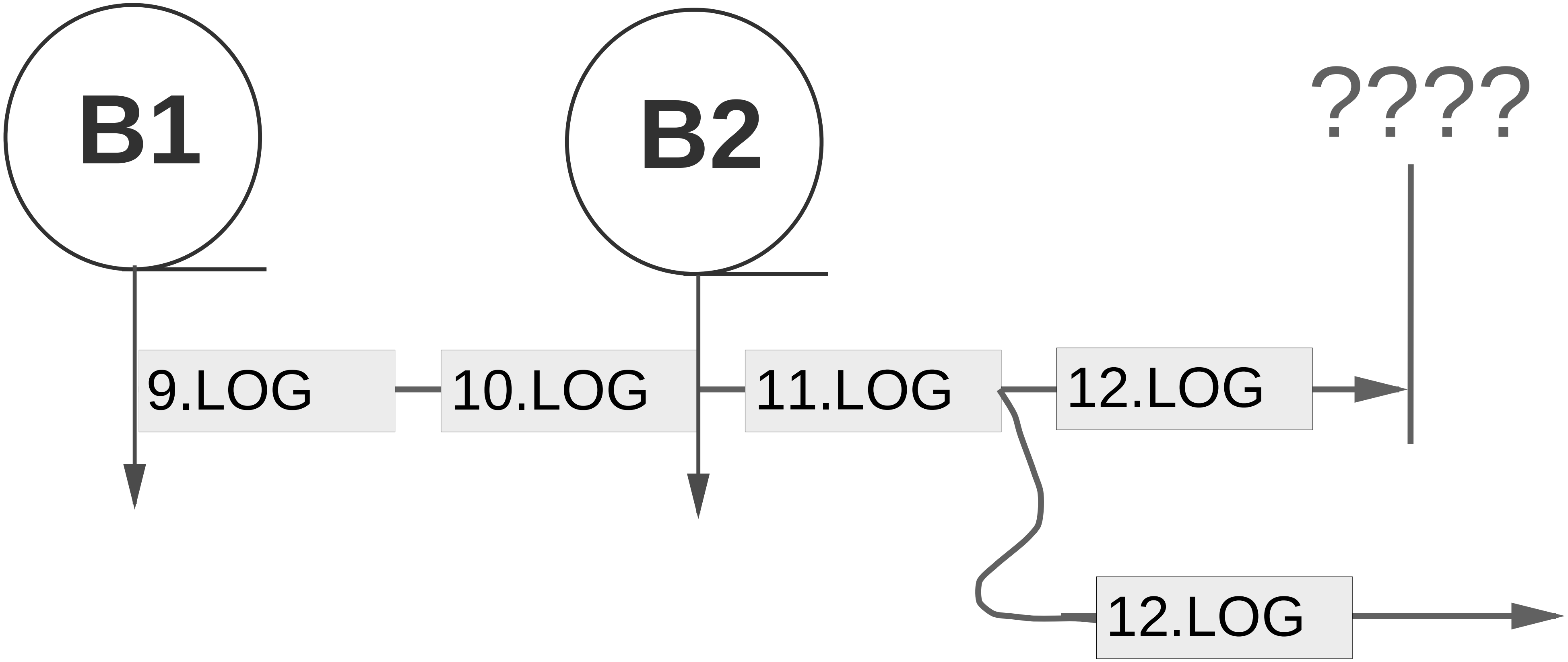
Parms: BACKUP Phase 2

- **OPT:** *BACKUP*
- **Parm1:** *D:*
- **Parm2:** *FRTEST*
- **Parm3:** *D:\SQLDBDIR\SQL00001.BKP*
- **Parm4:** *FRTEST-701479476*
- **Parm5:** *2*

Operational Considerations

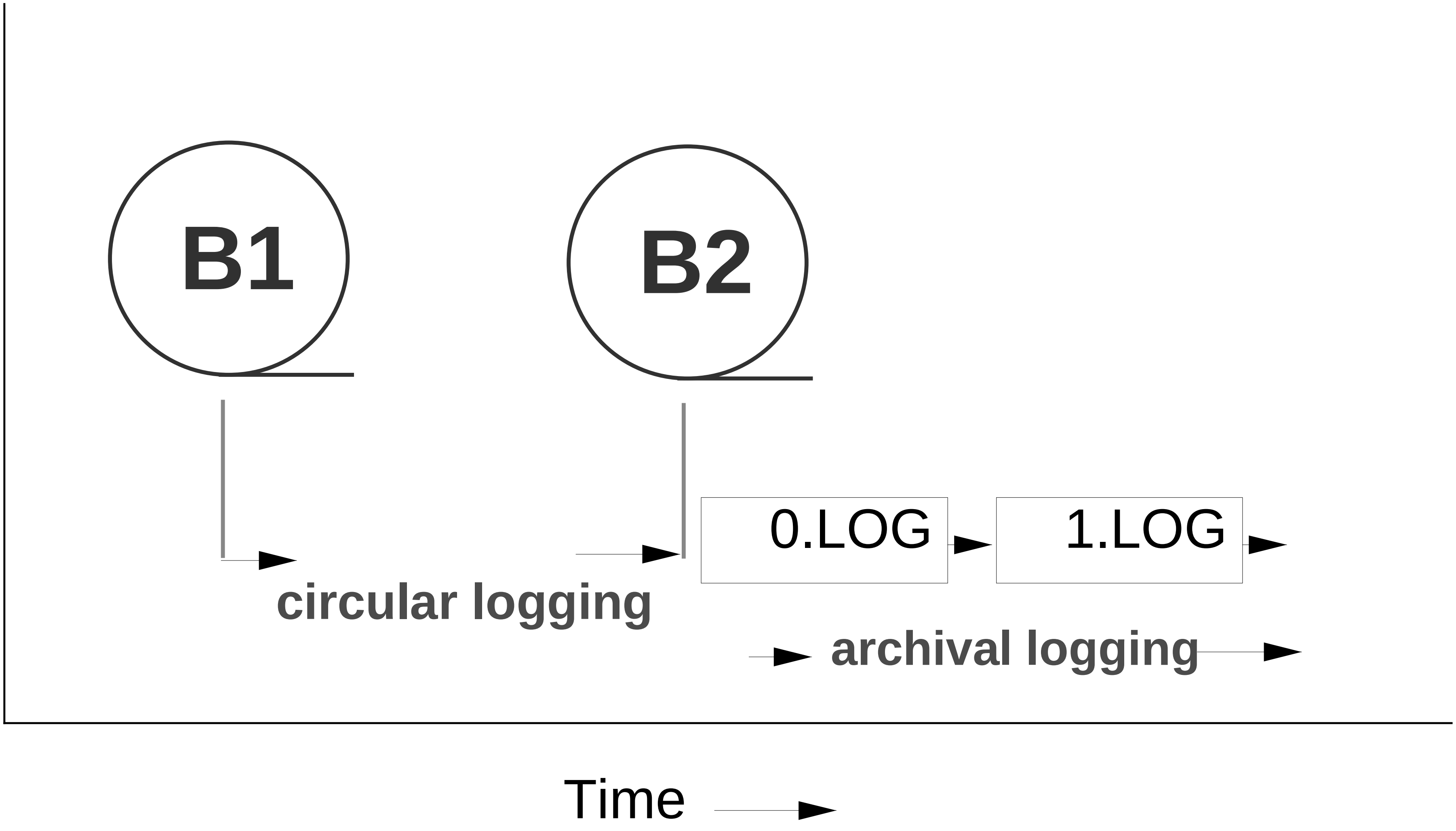


Multiple Recovery Paths



IBM DB2/2™

Non-recoverable backups



IBM DB2/2™

Coding Language

- **REXX**
  - interpretive
    - *99% of time will be to the device anyway*
  - ship source code by using rexx
  - not tamperproof
  - samples provided
  - easy to code

- **C, COBOL, or other 3GL**
  - compiled
  - do not ship source code
  - tamperproof exe
  - no samples provided
  - much harder to code

SQLUEXIT Coding

- **Coding Challenges:**
  - – Handling previously backed up databases via multiple directories or volumes
  - – Non-trivial coding
  - – Documentation essential for Operations personnel
- **Audit Log**
  - – Not necessary for recovery, but helpful
  - – Not backed up - would be lost along with system
  - – Needs more detail than the samples

Same EXIT: Multiple Databases

- **All enabled Databases will use the same User Exit program**
  - SQLUEXIT could call other programs
  - Choice of path could be controlled by a config file
  - Possible different devices used by different DBs
- **Risky to mix data on same media**
  - Depends on media type & software chosen
  - Will require a much more complex SQLUEXIT
- **Suggestion:**
  - 1 backup per removable media only!!

User Exit Destinations

- **Any dest that can called by command or API:**
  - 2.3gb Tape w/Sytos
  - Rewritable Optical Drive
  - IBM Internal Tape Drive w/PMTAPE
  - Lan Redirected Drive
  - Local Drive
  - TSO Account
  - Other OEMs
- **Any combination of the above**

Backup/Archive Destinations

- **2.3gb Tape w/Sytos**
  - Overhead of tape searches
  - Enormous capacity
  - Allows compression

Backup/Archive Destinations

- **Rewritable Optical Drive**
  - Small (127mb) size of optical disk
  - Very quick XCOPY
- **IBM Internal Tape Drive w/PMTAPE**
  - Small (120mb) size of tape cartridge
  - Fairly quick
  - Allows compression

Backup/Archive Destinations

- **Lan Redirected Drive**
  - Availability
  - Quick XCOPY
- **Local Drive**
  - Non-removable
  - Perishable
  - Head movement overlap

Backup/Archive Destinations

- **MVS TSO Account**
  - Access to network and TSO
  - Availability
  - Superior backup facilities of MVS
  - EHALLAPI interface from REXX:
    - LOGON account
    - MVS archive command
  - Also consider VM or AS/400 account

Dual Destinations

- **Any combination of the above**
  - Dual archiving increases likelihood of recovery
  - Syncronization

Tuning

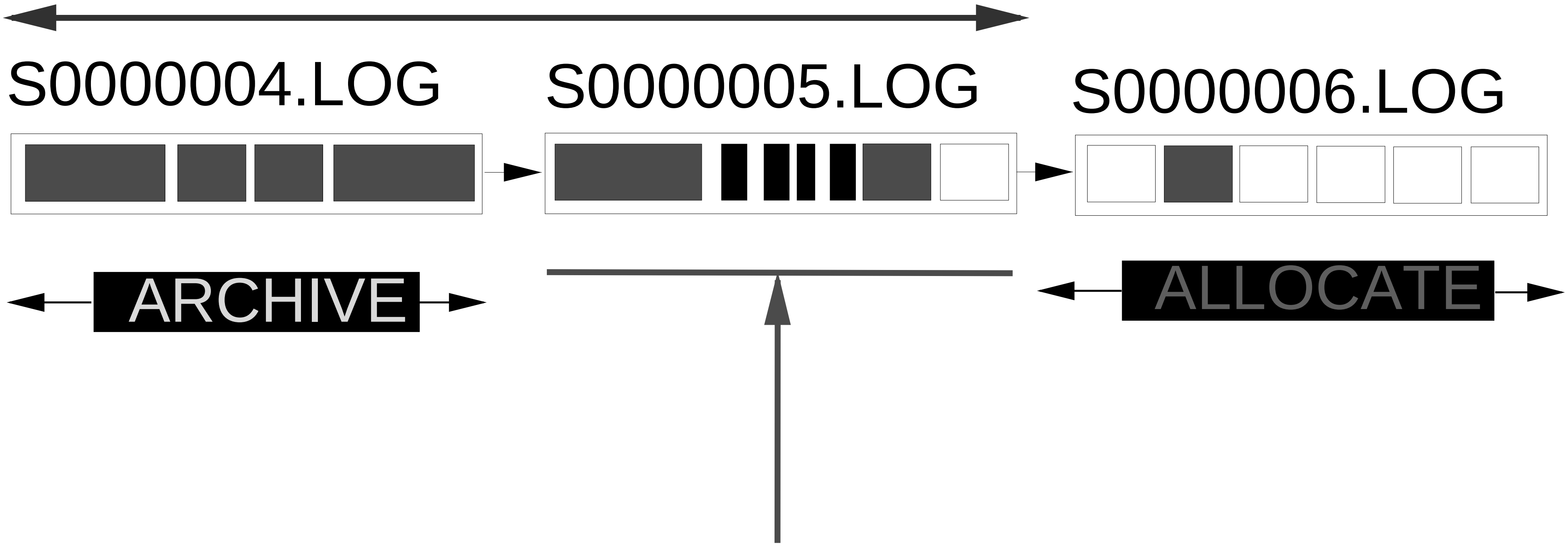
Archive Frequency

- **Use Audit Logs to determine how frequently ARCHIVE is running**
- **Adjust size of logs:**
  - CPU overhead of User Exit device may effect end use
  - Small log files & frequent archiving could overrun the device
  - Possibly archive several logs in one pass

Archive Frequency

- **Seek out a balance point:**
  - Archiving TOO FREQUENTLY
    - Archive software takes CPU cycles away
    - Contention for archival device
  - Archiving TOO INFREQUENTLY
    - Risk of loosing business data

Log Requirements



Average required Log space

S0000004.LOG

S0000005.LOG

S0000006.LOG

ARCHIVE

ALLOCATE

Not enough!

DB2/2™

Logs: How Many & How Big

- **No formula for computing**
  - Depends entirely on nature of Application
- **Value determined by**
  - testing
  - observation
  - knowledge of your application
- **Indicator: frequent LOG FULL error (SQLCODE -964C)**
  - increase LOGPRIMARY and/or LOGFILSZ

Demo

Demo Script

- **Create Database**
- **Change Parms: Log_Retain & User_Exit**
- **Take required backup**
  - note execution of user exit (twice)
- **Create tables**
  - note execution of user exit
- **Load tables**
  - note execution of user exit
- **Restore & roll forward to end of logs**
  - note execution of user exit program (multiple times)

IBM DB2/2™

Demo Programs

- The demo programs are all Copyright IBM Corporation 1993. Thi means that you can use them to learn about and experiment wit Forward Recovery. You'll also find them useful for learning abou REXX. However, the copyright means that you cannot sell them use them in any venture other than those explicitly named abov
- See the comments attached to each program. While the code ha been tested in a "demo" environment, we cannot be responsible for any code defects, bugs, shortcomings. You should thorough test them before using them on your own system.
- FRDEMO.CMD
- MON_LOGS.CMD
- MON_DB.CMD
- SQLUEXIT.CMD