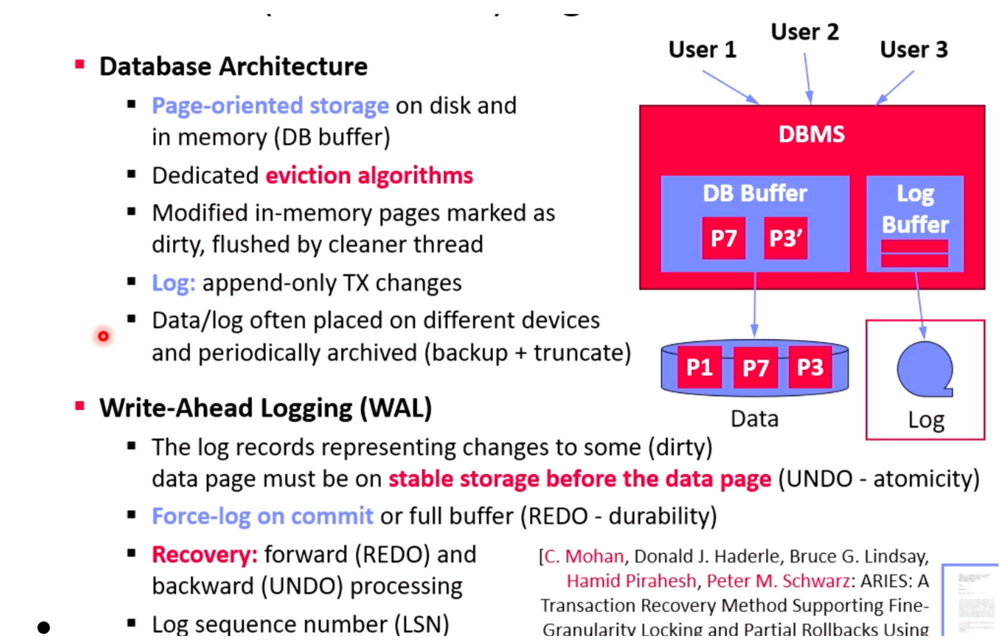
## Failure Types

- transaction failures
  - violated integrity constraints
  - R1-Recovery: partial undo of uncommitted tx
- system failures
  - HW/OS system crash, power outage, ...
  - kill all current TX
  - does not lose persistent data
  - R2-Recovery: partial redo of committed TX
  - R3-Recovery: global undo of uncommitted TX
- media failue
  - hard disk errors (non-restorable)
  - lose persistent data ==> need back up
  - R4-Recovery: global redo of all committed TX

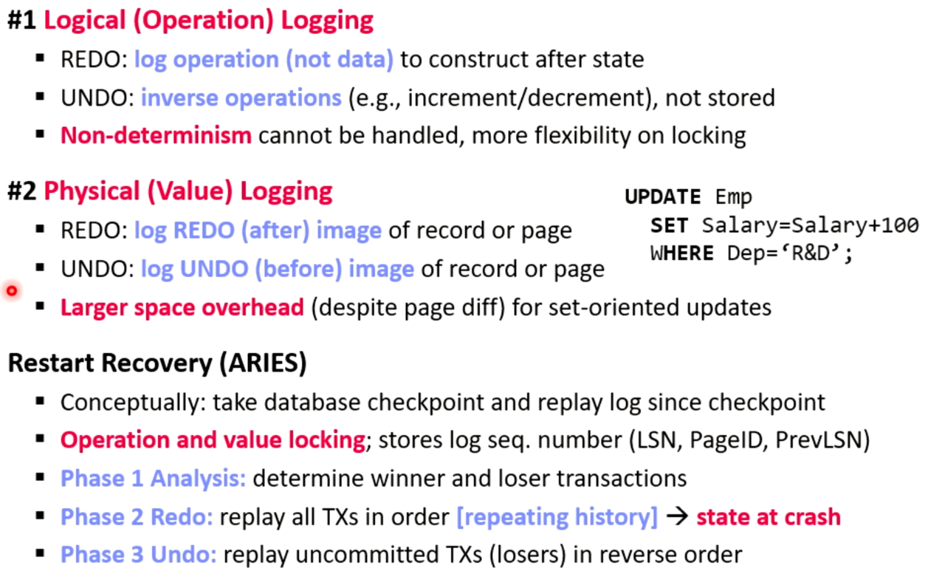
## Database Transaction Log



- **Database Architecture**
  - **Page-oriented storage** on disk and in memory (DB buffer)
  - Dedicated **eviction algorithms**
  - Modified in-memory pages marked as dirty, flushed by cleaner thread
  - **Log:** append-only TX changes
  - Data/log often placed on different devices and periodically archived (backup + truncate)
- **Write-Ahead Logging (WAL)**
  - The log records representing changes to some (dirty) data page must be on **stable storage before the data page** (UNDO - atomicity)
  - **Force-log on commit** or full buffer (REDO - durability)
  - **Recovery:** forward (REDO) and backward (UNDO) processing
  - Log sequence number (LSN)

[C. Mohan, Donald J. Haderle, Bruce G. Lindsay, Hamid Pirahesh, Peter M. Schwarz: ARIES: A Transaction Recovery Method Supporting Fine-Granularity Locking and Partial Rollbacks Using

## Logging Types



**#1 Logical (Operation) Logging**

- REDO: log operation (not data) to construct after state
- UNDO: inverse operations (e.g., increment/decrement), not stored
- **Non-determinism** cannot be handled, more flexibility on locking

**#2 Physical (Value) Logging**

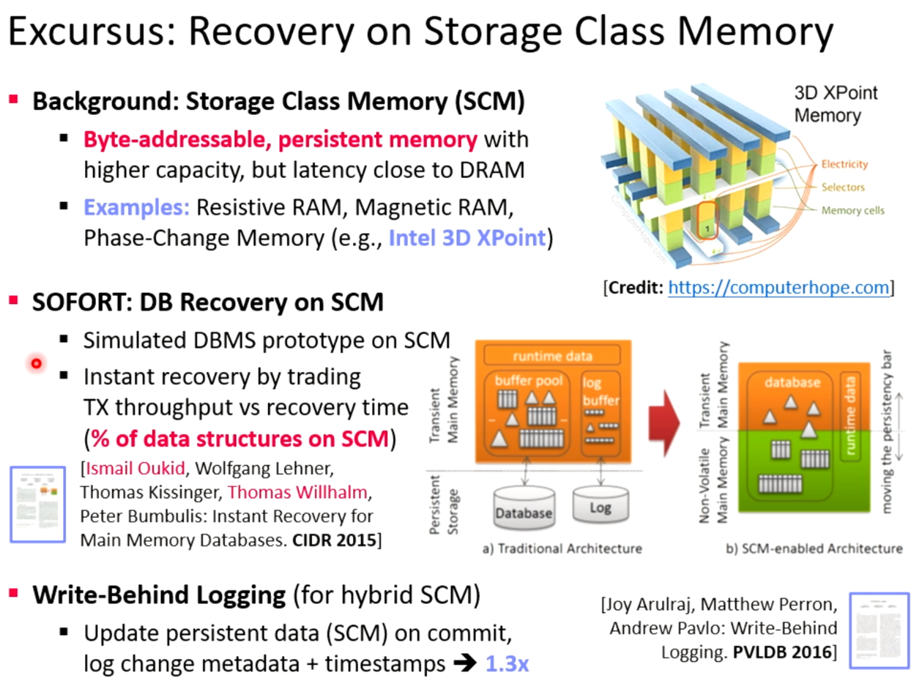
- REDO: log REDO (after) image of record or page
- UNDO: log UNDO (before) image of record or page
- **Larger space overhead** (despite page diff) for set-oriented updates

```
UPDATE Emp
  SET Salary=Salary+100
  WHERE Dep='R&D';
```

**Restart Recovery (ARIES)**

- Conceptually: take database checkpoint and replay log since checkpoint
- **Operation and value locking**; stores log seq. number (LSN, PageID, PrevLSN)
- Phase 1 Analysis: determine winner and loser transactions
- Phase 2 Redo: replay all TXs in order [repeating history] → **state at crash**
- Phase 3 Undo: replay uncommitted TXs (losers) in reverse order

## Recovery on Storage Class Memory



### Excursus: Recovery on Storage Class Memory

- **Background: Storage Class Memory (SCM)**
  - **Byte-addressable, persistent memory** with higher capacity, but latency close to DRAM
  - Examples: Resistive RAM, Magnetic RAM, Phase-Change Memory (e.g., Intel 3D XPoint)

3D XPoint Memory
Electricity
Selectors
Memory cells

[**Credit:** https://computerhope.com]

- **SOFORT: DB Recovery on SCM**
  - Simulated DBMS prototype on SCM
  - Instant recovery by trading TX throughput vs recovery time (**% of data structures on SCM**)

[Ismail Oukid, Wolfgang Lehner, Thomas Kissinger, Thomas Willhalm, Peter Bumbulis: Instant Recovery for Main Memory Databases. **CIDR 2015**]

- **Write-Behind Logging** (for hybrid SCM)
  - Update persistent data (SCM) on commit, log change metadata + timestamps → 1.3x

[Joy Arulraj, Matthew Perron, Andrew Pavlo: Write-Behind Logging. **PVLDB 2016**]