

Physical Storage Media

**These slides are a modified version of the slides of the book
“Database System Concepts, 5th Ed., [McGraw-Hill](#),
by Silberschatz, Korth and Sudarshan.
Original slides are available at www.db-book.com**

Resilience

A system is designed to provide a certain service

Resilience is the ability of a system to deliver a correct service

If the system stops delivering the intended service, we call this a failure.

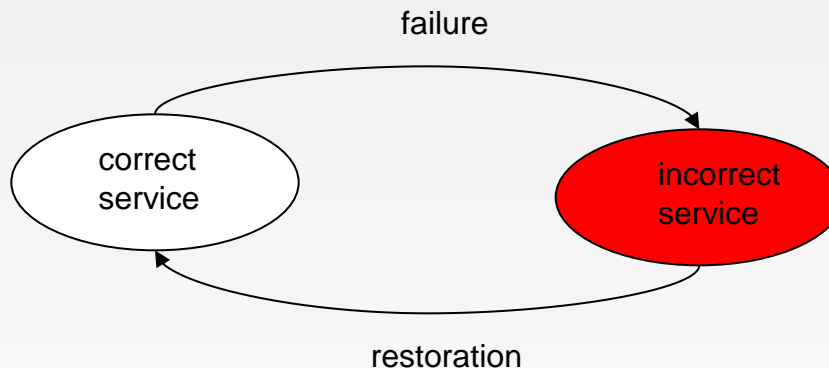
Threats : Failures, Errors and Faults

Correct service is delivered when the service implements the system function.

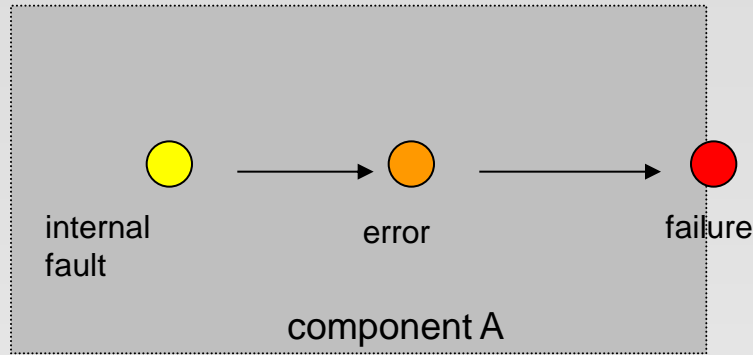
A **service failure**, often abbreviated failure, is an event that occurs when the delivered service deviates from correct service. A service fails either because it does not comply with the functional specification, or because this specification did not adequately describe the system function.

Failure is a transition from correct service to incorrect service

Restoration is the transition from incorrect service to correct service.



Threats : Failures, Errors and Faults

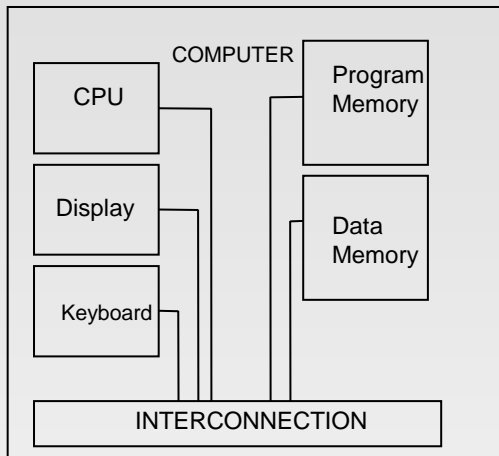


A fault causes an error in the internal state of the system. The error causes the system to fail

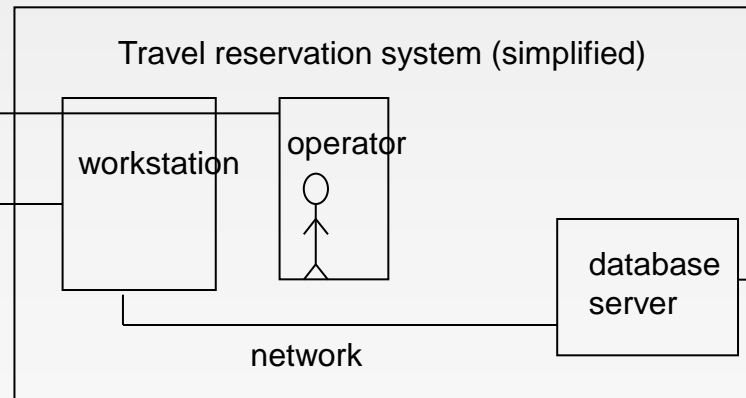
Partial failure: Services implementing the functions may leave the system in a degraded mode that still offers a subset of needed services to the user. The specification may identify several such modes, e.g., slow service, limited service, emergency service, etc. Here, we say that the system has suffered a partial failure of its functionality or performance.

Systems and components

A system is made out of components. Each component is a system in its own right



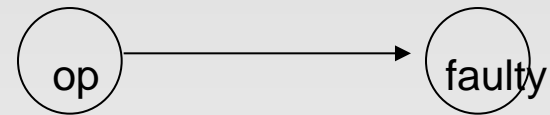
interface with
customer



interface with
airlines

Resilience evaluation

Reliability - $R(t)$, is the probability that the system performs correctly throughout the interval of time $[0, t]$, given that the system was performing correctly at time 0



Failure rate - The failure rate is the expected number of failures of a type of device per a given time period (e.g. $\lambda = 1/1000$, one failure per 1000 hours)

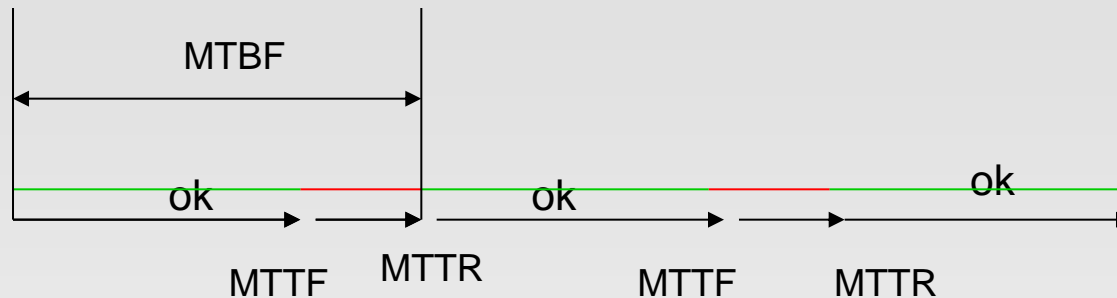
MTTF – The Mean Time To Failure is the expected time that a system will operate before the first failure occurs (e.g., 1000 hours)

MTTR - The Mean Time To Repair is the average time required to repair the system. MTTR is expressed in terms of a repair rate μ which is the average number of repairs that occur per time period, generally number of repairs per hours



Maintenability - $M(t)$ is the conditional probability that the system is repaired throughout the interval of time $[0, t]$, given that the system was faulty at time 0

Availability - $A(t)$, is the probability that the system is operating correctly and is available to perform its functions at the instant of time t .



Mean Time Between Failures - The MTBF is the average time between failures of the system, including the time required to repair the system and place it back into an operational status

$$\text{MTBF} = \text{MTTF} + \text{MTTR}$$

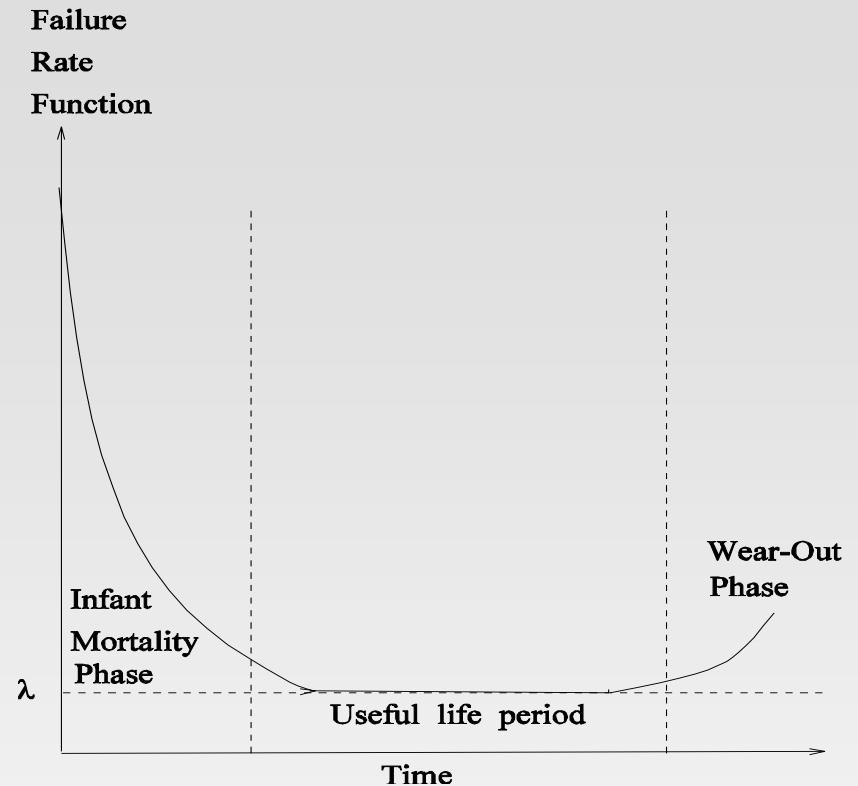
Hardware Reliability

rate at which failures occur in time

λ constant > 0 in the
useful life period

Failure rate λ (failures for million hours)
assume one failure every 2000 hours:

$$\lambda = 1/2000$$

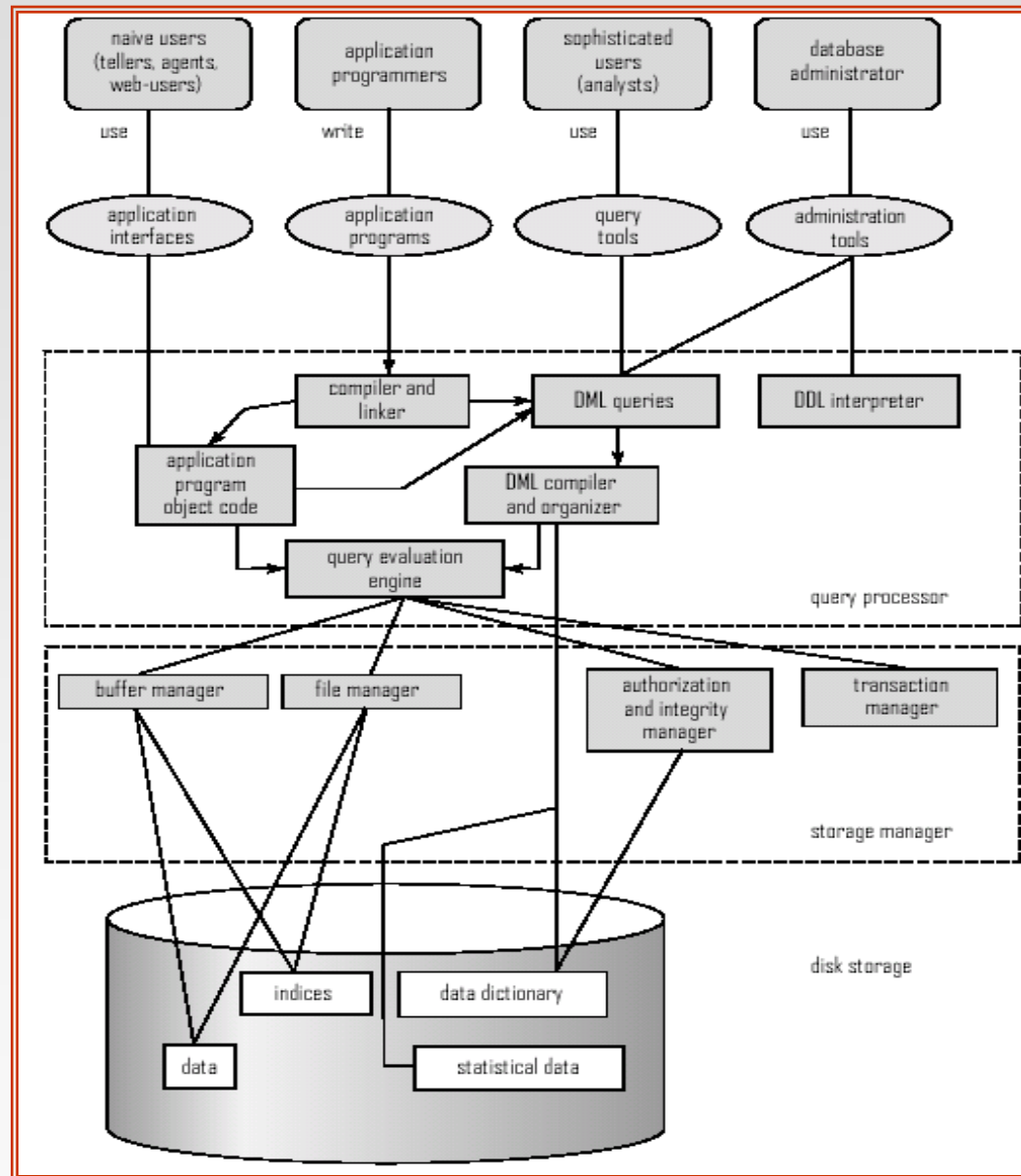


During early life, there is a higher failure rate, called infant mortality, due to the failures of weaker components. Often these infant mortalities result from defect or stress introduced in the manufacturing process.

Once the infant mortalities are eliminated, the system settles into operational life, in which the failure rate is approximately constant.

The system then approaches wear-out in which time and use cause the failure rate to increase.

DBMS: Overall Structure



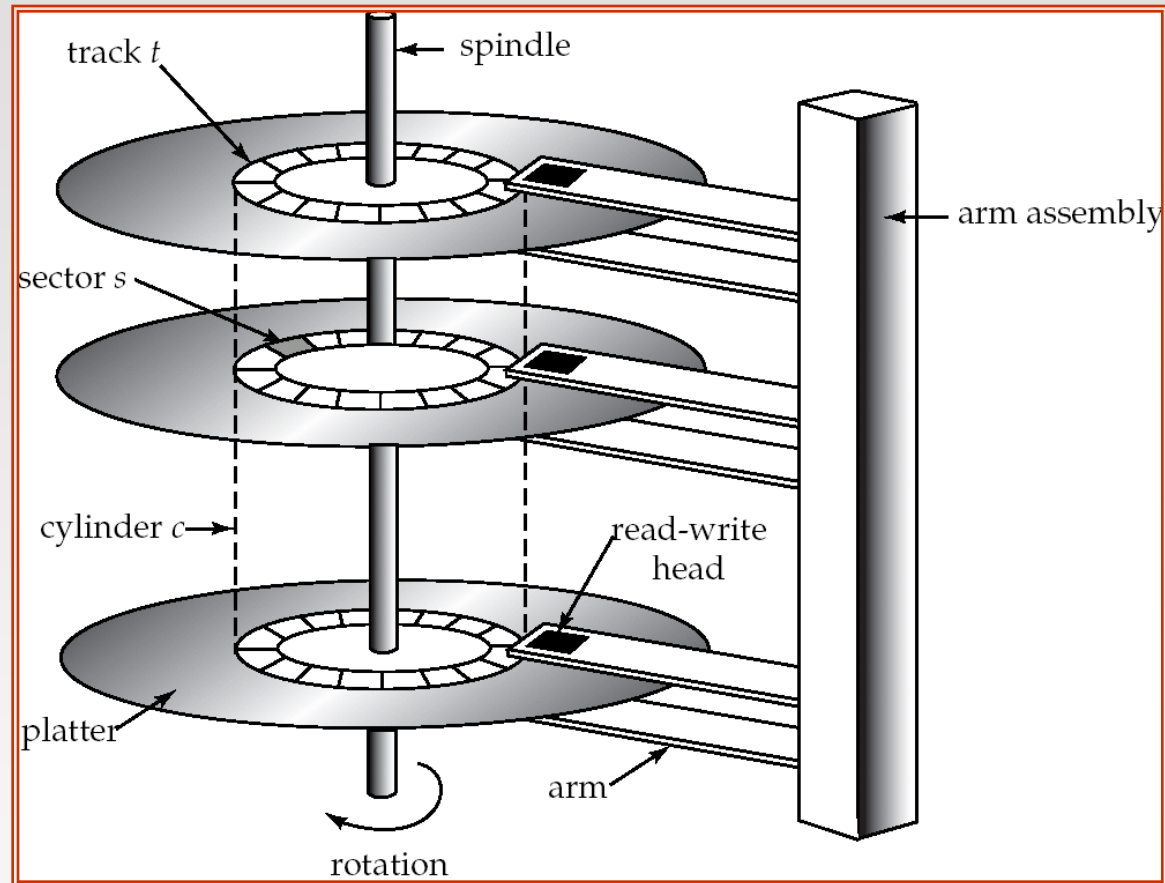
Physical Storage Media

■ Magnetic-disk

primary medium for the long term on-line storage of data

- Data is stored on spinning disk, and read/written magnetically
- Primary medium for the long-term storage of data; typically stores entire database.
- Data must be moved from disk to main memory for access, and written back for storage
 - ▶ Much slower access than main memory (more on this later)
- **direct-access** – possible to read data on disk in any order, unlike magnetic tape
- Capacities range up to roughly 400 GB currently
 - ▶ Growing constantly and rapidly with technology improvements (factor of 2 to 3 every 2 years)
- Survives power failures and system crashes
 - ▶ disk failure can destroy data

Magnetic Hard Disk Mechanism



NOTE: Diagram is schematic, and simplifies the structure of actual disk drives

Magnetic Disks

- **Read-write head**
 - Positioned very close to the platter surface (almost touching it)
 - Reads or writes magnetically encoded information.
- Surface of platter divided into circular **tracks**
 - Over 50K-100K tracks per platter on typical hard disks
- Each track is divided into **sectors**.
 - A sector is the smallest unit of data that can be read or written.
 - Sector size typically 512 bytes
 - Typical sectors per track: 500 (on inner tracks) to 1000 (on outer tracks)
- To read/write a sector
 - disk arm swings to position head on right track
 - platter spins continually; data is read/written as sector passes under head
- Head-disk assemblies
 - multiple disk platters on a single spindle (1 to 5 usually)
 - one head per platter, mounted on a common arm.
- **Cylinder** i consists of i^{th} track of all the platters

Magnetic Disks: reliability

■ Head crashes can be a problem

Earlier generation disks were susceptible to head-crashes

- Surface of earlier generation disks had metal-oxide coatings which would disintegrate on head crash and damage all data on disk
- Current generation disks are less susceptible to such disastrous failures, although individual sectors may get corrupted

Disk controller – interfaces between the computer system and the disk drive hardware.

- accepts high-level commands to read or write a sector
- initiates actions such as moving the disk arm to the right track and actually reading or writing the data

Magnetic Disks: reliability

Disk controller

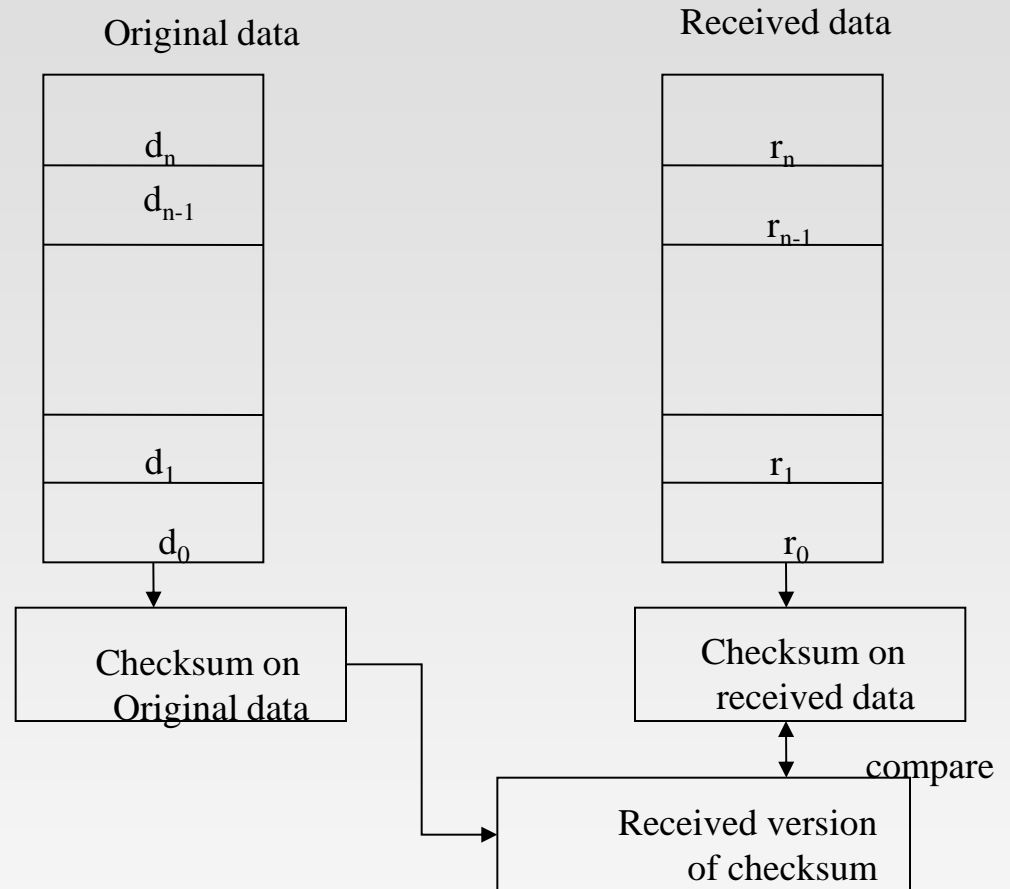
■ Read failure

Computes and attaches **checksums** to each sector to verify that data is read back correctly

- ▶ If data is corrupted, with very high probability stored checksum won't match recomputed checksum
- Ensures successful writing by reading back sector after writing it
- Performs **remapping of bad sectors**
maps a bad sector to a different physical location
(when the disk is formatted or when an attempt is made to write the sector)

Checksums

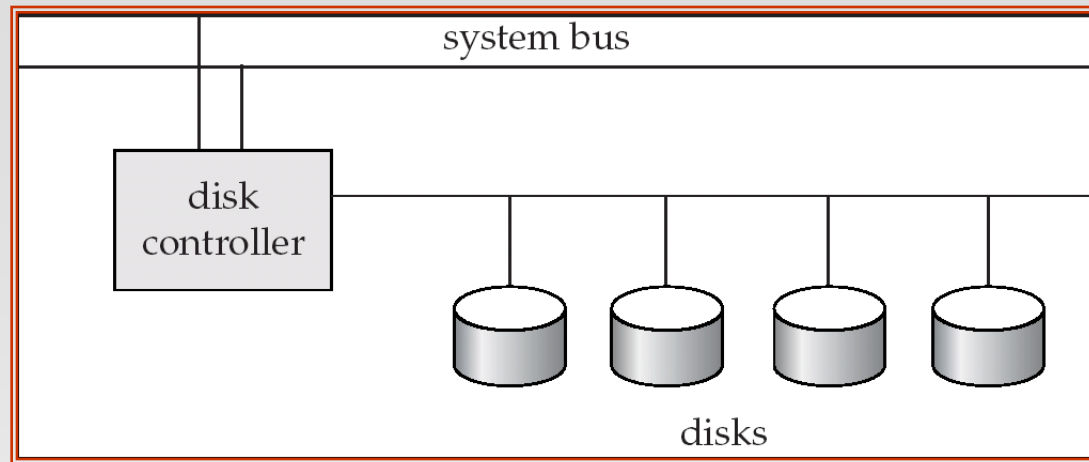
- when blocks of data must be transferred (data transfer between mass-storage device) checksum is added to the block of data to achieve error detection
- checksum is basically the sum of the original data
- checksum for a block of s words is formed by adding together all of the words in the block modulo- n , where n is arbitrary.



Disks organization

- Disks can be
 - 1) connected directly to the disk interface of the computer system
or
 - 2) situated remotely and connected by a high speed network to the disk controller interface (case of mainframes and servers)
- Remote access to disks means that
 - Disks can be shared by multiple computers that could run different parts of an application in parallel
 - Disks can be kept in a central server room where they are monitored

Disk Subsystem



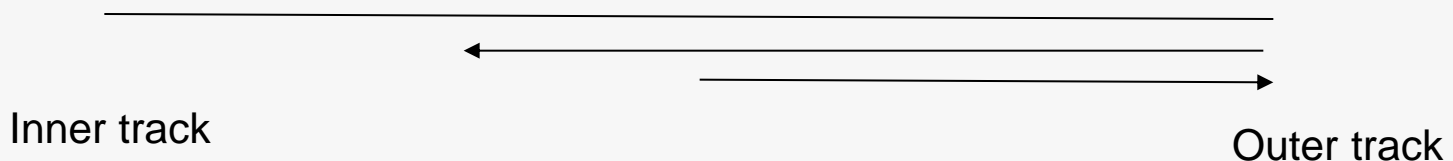
- Multiple disks connected to a computer system through a controller
- Disk interface standards families
 - E.g., **SCSI** (Small Computer System Interconnect) range of standards
 - Several variants of each standard (different speeds and capabilities)
- In particular, disks can be organized locally using a storage organization technique called **RAID: Redundant Arrays of Independent Disks** to give the server a **logical view of a “very large” and “reliable” disk**

Performance Measures of Disks

- **Access time** – the time it takes from when a read or write request is issued to when data transfer begins. Consists of:
 - **Seek time** – time it takes to reposition the arm over the correct track.
 - ▶ 4 to 10 milliseconds on typical disks
 - **Rotational latency** – time it takes for the sector to be accessed to appear under the head.
 - ▶ 4 to 11 milliseconds on typical disks (5400 to 15000 r.p.m.)
- **Data-transfer rate** – the rate at which data can be retrieved from or stored to the disk.
 - 25 to 100 MB per second max rate, lower for inner tracks

Optimization of Disk-Block Access

- **Block** – a contiguous sequence of sectors from a single track
 - data is transferred between disk and main memory in blocks
 - sizes range from 512 bytes to several kilobytes
 - ▶ Smaller blocks: more transfers from disk
 - ▶ Larger blocks: more space wasted due to partially filled blocks
 - ▶ Typical block sizes today range from 4 to 16 kilobytes
- **Disk-arm-scheduling** algorithms order pending accesses to tracks so that disk arm movement is minimized
 - **elevator algorithm** : move disk arm in one direction (from outer to inner tracks or vice versa), processing next request in that direction, till no more requests in that direction, then reverse direction and repeat



Optimization of Disk Block Access (Cont.)

- **File organization** – optimize block access time by organizing the blocks to correspond to how data will be accessed
 - E.g. Store related information on the same or nearby cylinders.
 - Files may get **fragmented** over time
 - ▶ E.g. if data is inserted to/deleted from the file
 - ▶ Or free blocks on disk are scattered, and newly created file has its blocks scattered over the disk
 - ▶ Sequential access to a fragmented file results in increased disk arm movement
 - Some systems have utilities to **defragment** the file system, in order to speed up file access

RAID

■ RAID: Redundant Arrays of Independent Disks

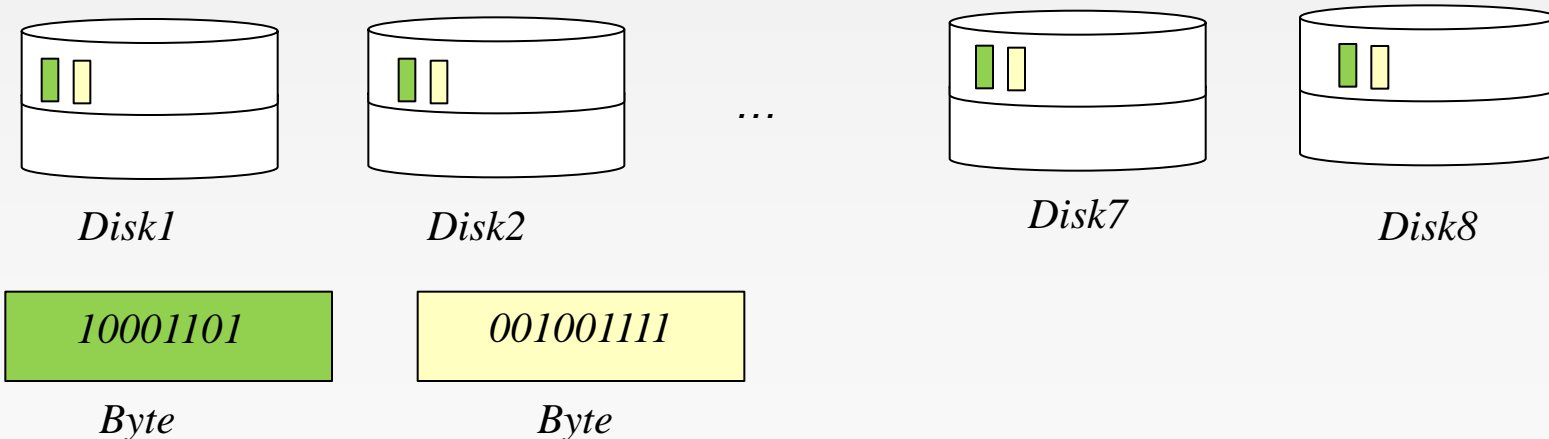
- disk organization techniques that manage a large numbers of disks, providing a view of a single disk of
 - ▶ **high capacity** and **high speed** by using multiple disks in parallel, and
 - ▶ **high reliability** by storing data redundantly, so that data can be recovered even if a disk fails
- The chance that some disk out of a set of N disks will fail is much higher than the chance that a specific single disk will fail.
 - E.g., a system with 100 disks, each with MTTF of 100,000 hours (approx. 11 years), will have a system MTTF of 1000 hours (approx. 41 days)
 - Techniques for using redundancy to avoid data loss are critical with large numbers of disks
- Originally a cost-effective alternative to large, expensive disks
 - I in RAID originally stood for “inexpensive”
 - Today RAIDs are used for their higher reliability and bandwidth.
 - ▶ The “I” is interpreted as independent

Improvement of Reliability via Redundancy

- **Redundancy** – store extra information that can be used to rebuild information lost in a disk failure
- E.g., **Mirroring** (or **shadowing**)
 - Duplicate every disk. Logical disk consists of two physical disks.
 - Every write is carried out on both disks
 - ▶ Reads can take place from either disk
 - If one disk in a pair fails, data still available in the other
 - ▶ Data loss would occur only if a disk fails, and its mirror disk also fails before the system is repaired
 - Probability of combined event is very small
 - » Except for dependent failure modes such as fire or building collapse or electrical power surges

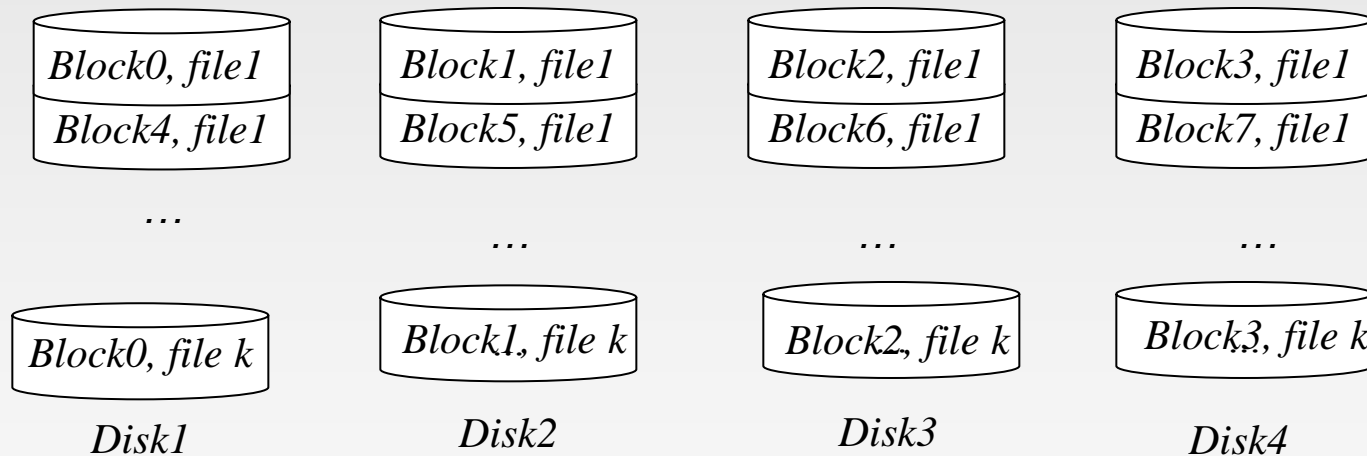
Improvement in Performance via Parallelism

- Two main goals of parallelism in a disk system:
 1. Load balance multiple small accesses to increase throughput
 2. Parallelize large accesses to reduce response time.
- Improve transfer rate by striping data across multiple disks.
- **Bit-level striping** – split the bits of each byte across multiple disks
 - In an array of eight disks, write bit i of each byte to disk i .
 - Each access can read data at eight times the rate of a single disk.
 - But seek/access time worse than for a single disk
 - ▶ Bit level striping is not used much any more



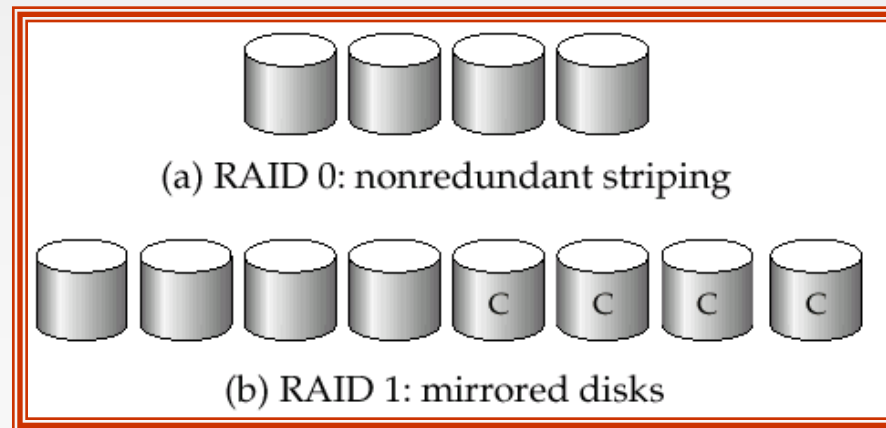
Improvement in Performance via Parallelism

- **Block-level striping** – with n disks, block i of a file goes to disk $(i \bmod n) + 1$
 - Requests for different blocks can run in parallel if the blocks reside on different disks
 - A request for a long sequence of blocks can utilize all disks in parallel



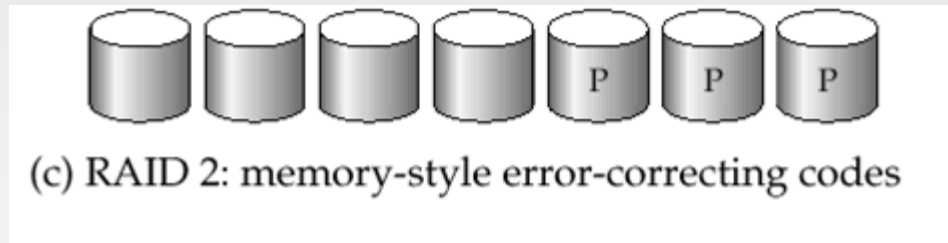
RAID Levels

- Schemes to provide redundancy at lower cost by using disk striping combined with parity bits
 - Different RAID organizations, or RAID levels, have differing cost, performance and reliability characteristics
- **RAID Level 0: Block striping; non-redundant.**
 - Used in high-performance applications where data loss is not critical.
- **RAID Level 1: Mirrored disks** with block striping
 - Offers best write performance.
 - Popular for applications such as storing log files in a database system.



RAID Levels (Cont.)

- **RAID Level 2: Memory-Style Error-Correcting-Codes** (ECC) with bit striping
- Each byte is assigned a parity bit: the bit records whether the number of bits in the byte that are set to 1 is even or odd
- If one bit in the byte gets damaged the parity of the byte changes and will not match the computed parity
ALL 1-BIT ERRORS ARE DETECTED
- Error correcting codes store extra bits to reconstruct the data if a single bit gets damaged
- Disks labelled P store the ECC



Example: odd parity

Byte 11010000

Disk1:0 Disk2:0 Disk3: 0 Disk4:0

more bits for error correction

Disk5:1 Disk6:0 Disk7:1 Disk8: 1 Parity code: 1

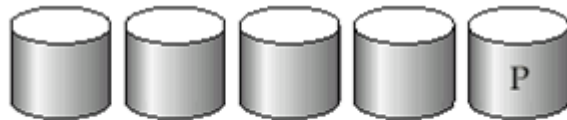
if one disk fails, the remaining bits and the ECC bit can be read by other disks

RAID Levels (Cont.)

■ RAID Level 3: Bit-Interleaved Parity

exploit the fact that disk controllers, unlike memory systems, can detect whether a sector has been read correctly

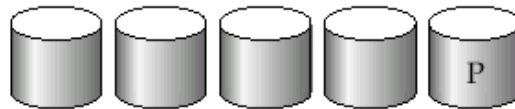
- a single parity bit is enough for error correction, not just detection, since we know which disk has failed
 - ▶ When writing data, corresponding parity bits must also be computed and written to a parity bit disk
 - ▶ To recover data in a damaged disk, compute the parity of the bits from the sectors in the other disks. If the parity is equal to the stored parity, the missing bit is 0; otherwise the missing bit is 1.
- Good as Level 2, but less expensive in the number of extra disks (one disk overhead)
- Benefits over Level 1: needs only one parity disk for several disks (Level 1, one mirror disk for every disk)



(d) RAID 3: bit-interleaved parity

RAID Levels (Cont.)

- **RAID Level 4: Block-Interleaved Parity**; uses block-level striping, and keeps a parity block on a separate disk for corresponding blocks from N other disks.
 - When writing data block, corresponding block of parity bits must also be computed and written to parity disk
 - To find value of a damaged block, compute parity of bits from corresponding blocks (including parity block) from other disks.

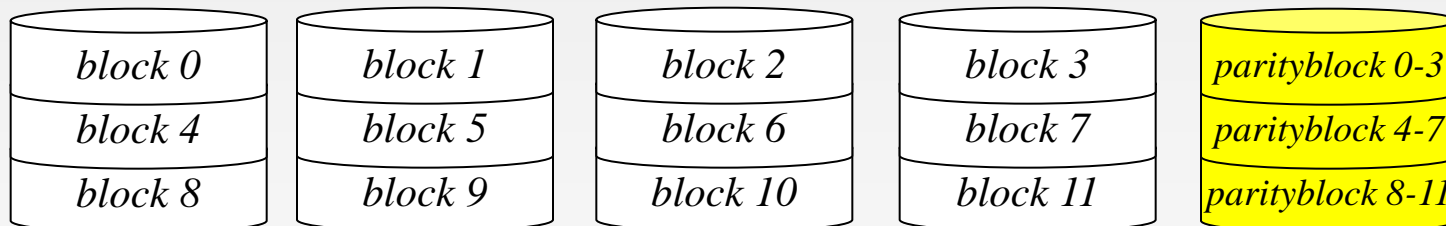


(e) RAID 4: block-interleaved parity

RAID Levels (Cont.)

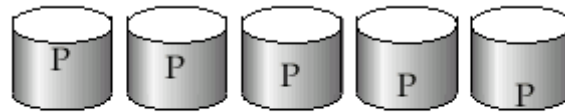
■ RAID Level 4 (Cont.)

- Before writing a block, parity data must be computed
 - ▶ Can be done by using old parity block, old value of current block and new value of current block (2 block reads + 2 block writes)
 - ▶ Or by recomputing the parity value using the new values of blocks corresponding to the parity block
- Parity block becomes a bottleneck for independent block writes since every block write also writes to parity disk



RAID Levels (Cont.)

- **RAID Level 5: Block-Interleaved Distributed Parity**; partitions data and parity among all $N + 1$ disks, rather than storing data in N disks and parity in 1 disk.
 - E.g., with 5 disks, parity block for n th set of blocks is stored on disk $(n \bmod 5) + 1$, with the data blocks stored on the other 4 disks.



(f) RAID 5: block-interleaved distributed parity

P0	0	1	2	3
4	P1	5	6	7
8	9	P2	10	11
12	13	14	P3	15
16	17	18	19	P4

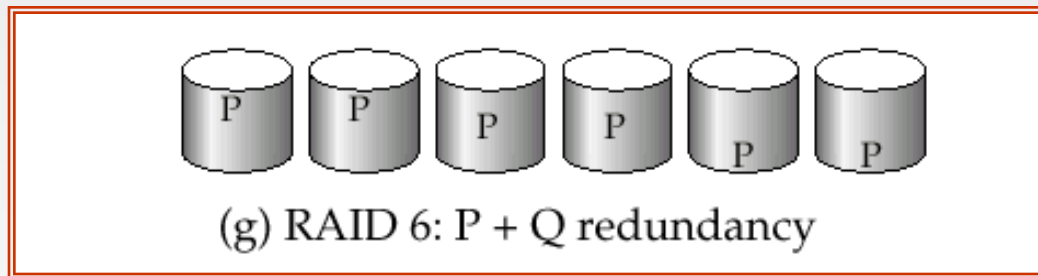
RAID Levels (Cont.)

■ RAID Level 5 (Cont.)

- For each set of N logical blocks, one of the disks store the parity and the other N disks store the blocks
- The P's are distributed across all the disks
- A parity block can not store parity for bocks of the same disk, since then a disk failure would result in loss of data as well as of parity(failure not recoverable)
- Level 5 subsumes Level 4

RAID Levels (Cont.)

- **RAID Level 6: P+Q Redundancy** scheme; similar to Level 5, but stores extra redundant information to guard against multiple disk failures.
 - Better reliability than Level 5 at a higher cost; not used as widely.
 - Level 6, instead of using parity, uses ECC.
 - In the figure 2 bits of redundant data are stored for every 4 bits of data and the system can tolerate two disk failures



Choice of RAID Level

- Factors in choosing RAID level
 - Monetary cost
 - Performance: Number of I/O operations per second, and bandwidth during normal operation
 - Performance during failure
 - Performance during rebuild of failed disk
 - ▶ Including time taken to rebuild failed disk
- RAID 0 is used only when data safety is not important
- Level 2 and 4 never used since they are subsumed by 3 and 5
- Level 3 is not used anymore since bit-striping forces single block reads to access all disks, wasting disk arm movement, which block striping (level 5) avoids
- Level 6 is rarely used since levels 1 and 5 offer adequate safety for almost all applications
- So competition is between 1 and 5 only

Stable Storage

A mythical form of storage that survives all failure

Approximated by maintaining multiple copies on distinct disk

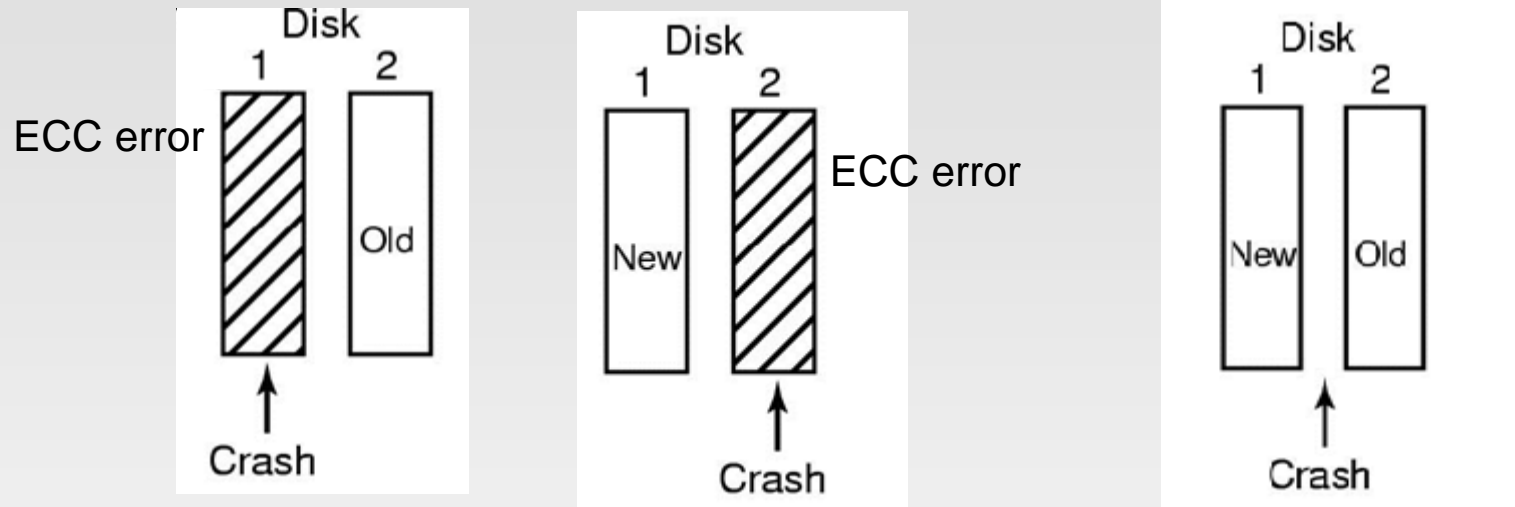
Often stable storage functionality is achieved by mirroring data on separate disks via RAID technology (level 1 or greater).

Protecting storage media from failure during data transfer(write())

- Write() executed in sequence on separate disks
- Assume two blocks on separate disks are inconsistent:
 - If either copy of an **inconsistent block** is detected to have an **error** (bad checksum), overwrite it by the other copy.
 - If both have no error, but **are different**, overwrite the second block by the first block

While one of those places might return an error, the other is statistically unlikely to do the same, resulting in a higher level of stable storage.

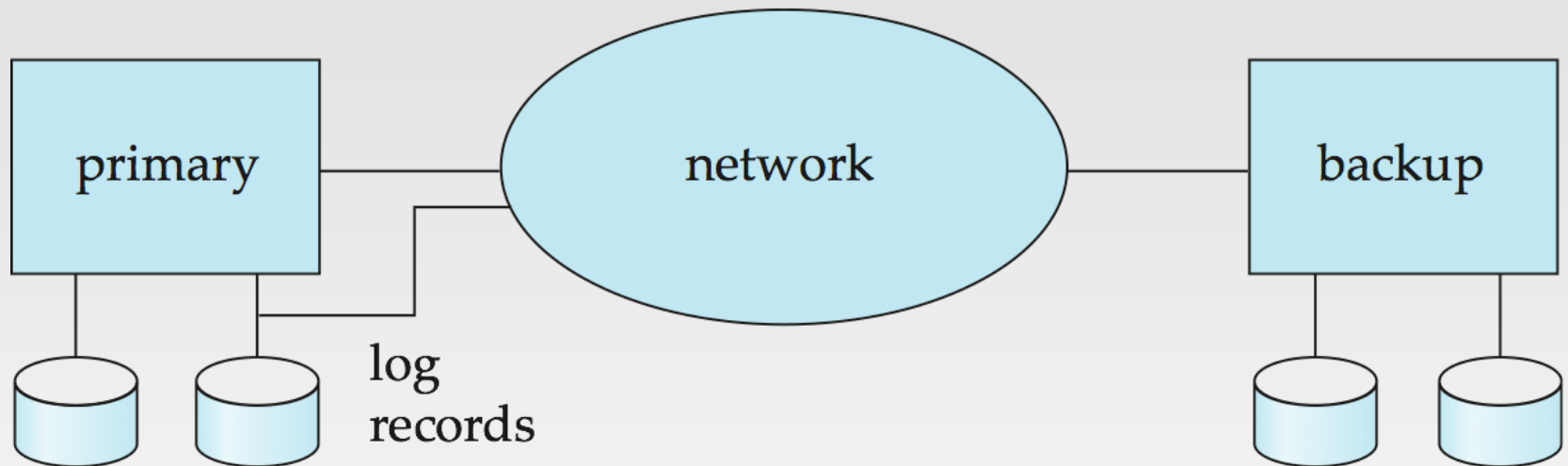
Crash in Stable Storage



Remote Backup Systems

Remote Backup Systems

- Remote backup systems provide high availability by allowing transaction processing to continue even if the primary site is destroyed.



Remote Backup Systems (Cont.)

- **Detection of failure:** Backup site must detect when primary site has failed
 - to distinguish primary site failure from link failure maintain several communication links between the primary and the remote backup.
 - Heart-beat messages
- **Transfer of control:**
 - To take over control backup site first perform recovery using its copy of the database and all the log records it has received from the primary.
 - ▶ Thus, completed transactions are redone and incomplete transactions are rolled back.
 - When the backup site takes over processing it becomes the new primary
 - To transfer control back to old primary when it recovers, old primary must receive redo logs from the old backup and apply all updates locally.

Remote Backup Systems (Cont.)

- **Time to recover:** To reduce delay in takeover, backup site periodically processes the redo log records (in effect, performing recovery from previous database state), performs a checkpoint, and can then delete earlier parts of the log.
- **Hot-Spare** configuration permits very fast takeover:
 - Backup continually processes redo log record as they arrive, applying the updates locally.
 - When failure of the primary is detected the backup rolls back incomplete transactions, and is ready to process new transactions.
- Alternative to remote backup: distributed database with replicated data
 - Remote backup is faster and cheaper, but less tolerant to failure

Remote Backup Systems (Cont.)

- Ensure durability of updates by delaying transaction commit until update is logged at backup; avoid this delay by permitting lower degrees of durability.
- **One-safe:** commit as soon as transaction's commit log record is written at primary
 - Problem: updates may not arrive at backup before it takes over.
- **Two-very-safe:** commit when transaction's commit log record is written at primary and backup
 - Reduces availability since transactions cannot commit if either site fails.
- **Two-safe:** proceed as in two-very-safe if both primary and backup are active. If only the primary is active, the transaction commits as soon as its commit log record is written at the primary.
 - Better availability than two-very-safe; avoids problem of lost transactions in one-safe.

Replication with Weak Consistency

- Many commercial databases support replication of data with weak degrees of consistency (i.e., without a guarantee of serializability)
- E.g.: **master-slave replication**: updates are performed at a single “master” site, and propagated to “slave” sites.
 - Propagation is not part of the update transaction: it is decoupled
 - ▶ May be immediately after transaction commits
 - ▶ May be periodic
 - Data may only be read at slave sites, not updated
 - ▶ No need to obtain locks at any remote site
 - Particularly useful for distributing information
 - ▶ E.g. from central office to branch-office
 - Also useful for running read-only queries offline from the main database

Replication with Weak Consistency (Cont.)

- Replicas should see a **transaction-consistent snapshot** of the database
 - That is, a state of the database reflecting all effects of all transactions up to some point in the serialization order, and no effects of any later transactions.
- E.g. Oracle provides a **create snapshot** statement to create a snapshot of a relation or a set of relations at a remote site
 - snapshot refresh either by recomputation or by incremental update
 - Automatic refresh (continuous or periodic) or manual refresh

Multimaster and Lazy Replication

- With multimaster replication (also called update-anywhere replication) updates are permitted at any replica, and are automatically propagated to all replicas
 - Basic model in distributed databases, where transactions are unaware of the details of replication, and database system propagates updates as part of the same transaction
 - ▶ Coupled with 2 phase commit
- Many systems support **lazy propagation** where updates are transmitted after transaction commits
 - Allows updates to occur even if some sites are disconnected from the network, but at the cost of consistency