Physical Design

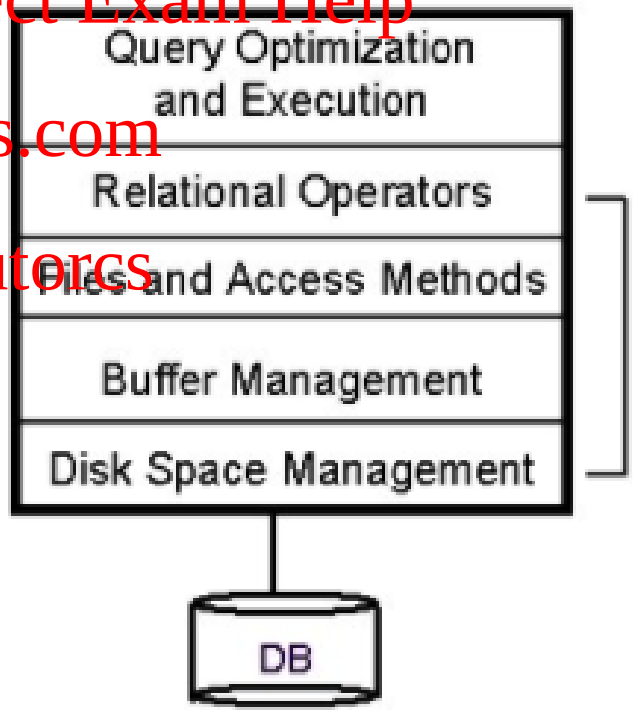
Week 4

Improving Database Performance

RAID

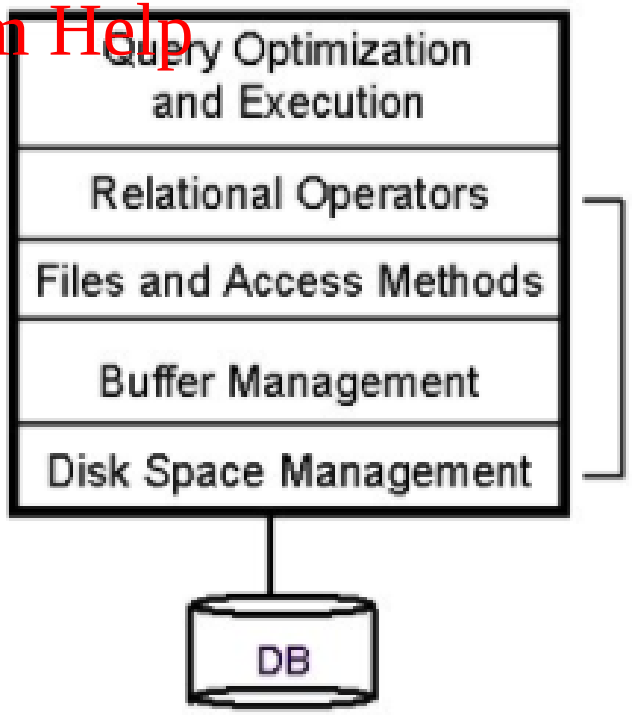
DBMS Architecture

- A typical DBMS has a layered architecture
- This is one of several possible architectures:
  - each system has its own variations



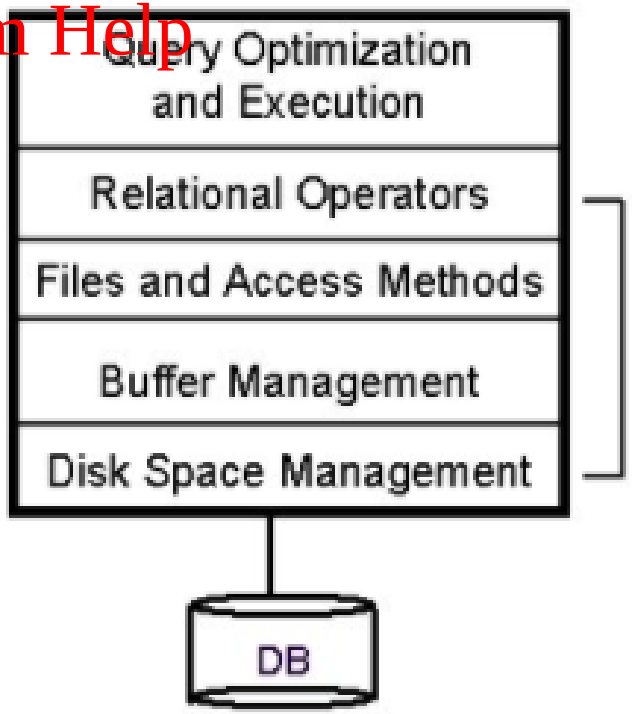
DBMS Architecture

- DBMS needs to retrieve, update and process
- persistently stored data
  - Data is huge
  - Must persist across executions
  - But has to be fetched into main memory when DBMS processes the data
- Storage consideration is an important factor in planning a database system (physical layer)



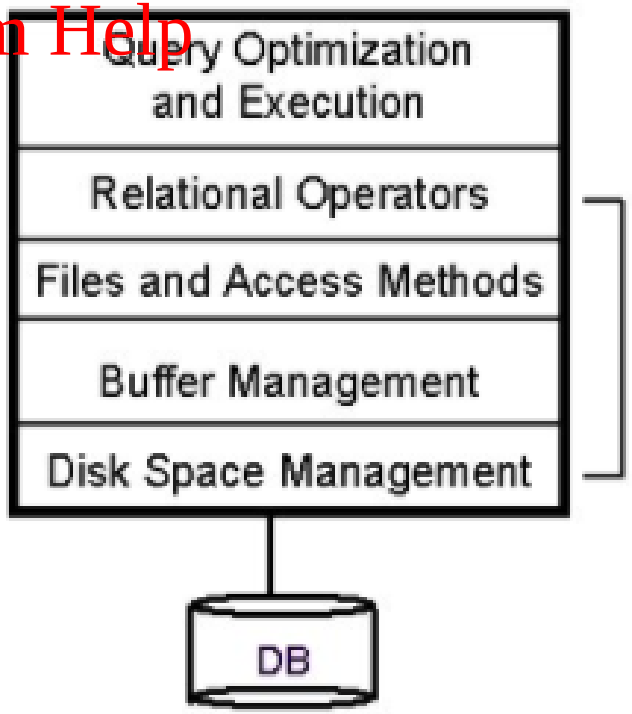
DBMS Architecture

- Data is stored on a storage medium
  - Media differ in terms of
    - **Random Access** Speed
      - Average time to access a random piece of data at a known media position
      - Usually measured in ms or ns
    - **Random/Sequential Read/Write** speed
    - **Capacity**
    - **Cost** per Capacity



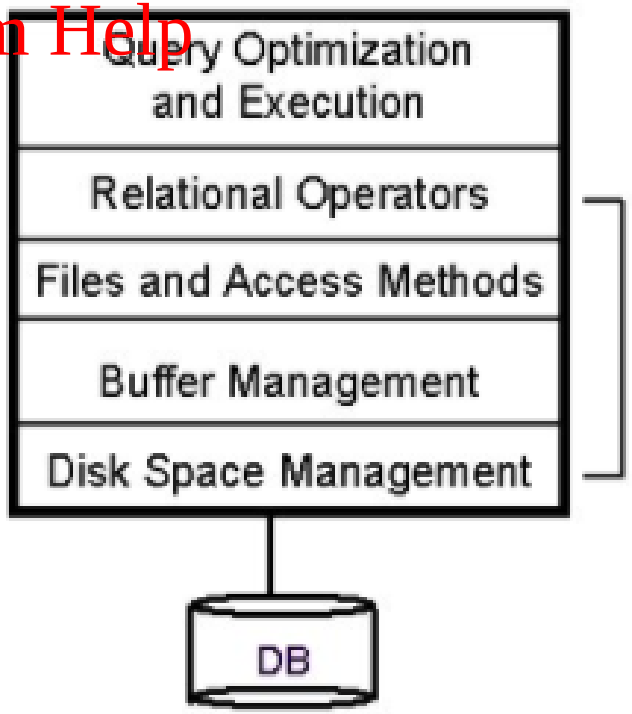
DBMS Architecture

- **Random/Sequential Read/Write** speed
- How fast an SSD can read/write one large continuous file
- Is data in sequential blocks or is it scattered in random blocks all over the drive?
- Transfer Rate:
  - Average amount of consecutive data which can be transferred per time unit
  - Usually measured in KB/sec, MB/sec, GB/sec,…
  - Sometimes also in Kb/sec, Mb/sec, Gb /sec



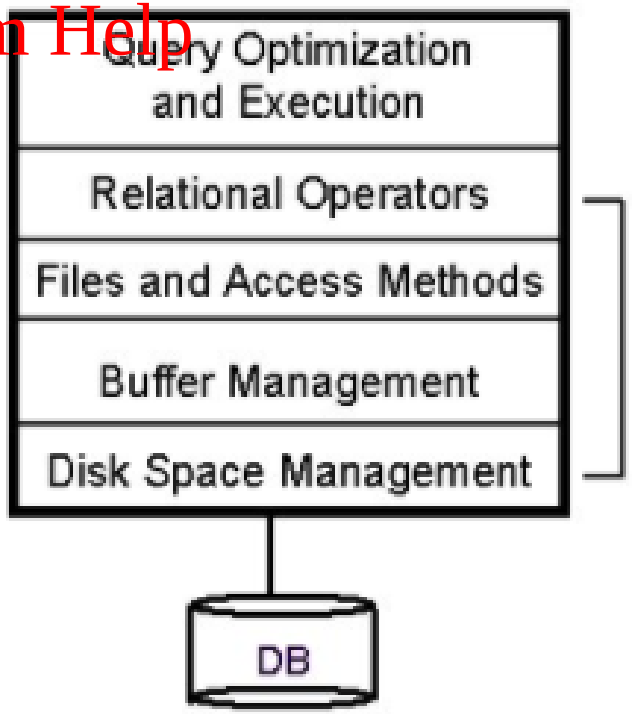
DBMS Architecture

- Capacity
  - Quantifies the amount of data which can be stored



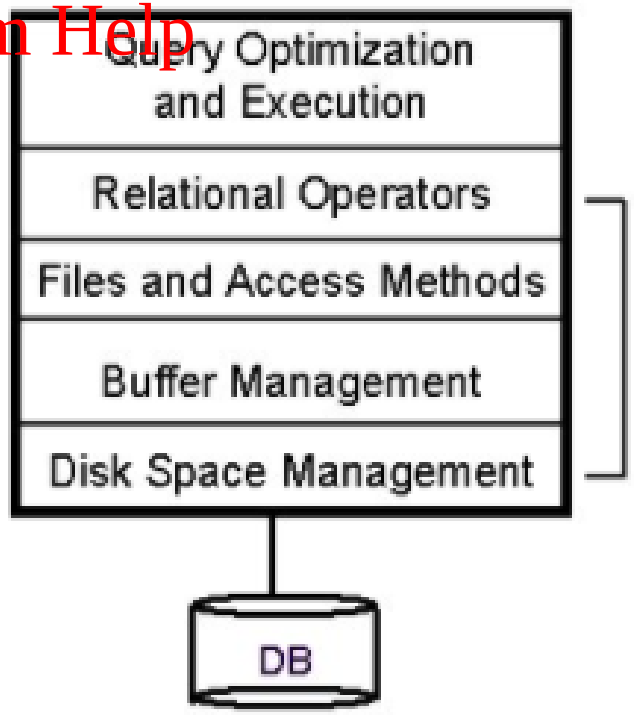
DBMS Architecture

- The unit of information for reading data from disk, or writing data to disk, is a **page**
- **Disks**: Can retrieve random page at fixed cost
  - Reading several consecutive pages is much cheaper than reading pages in random order



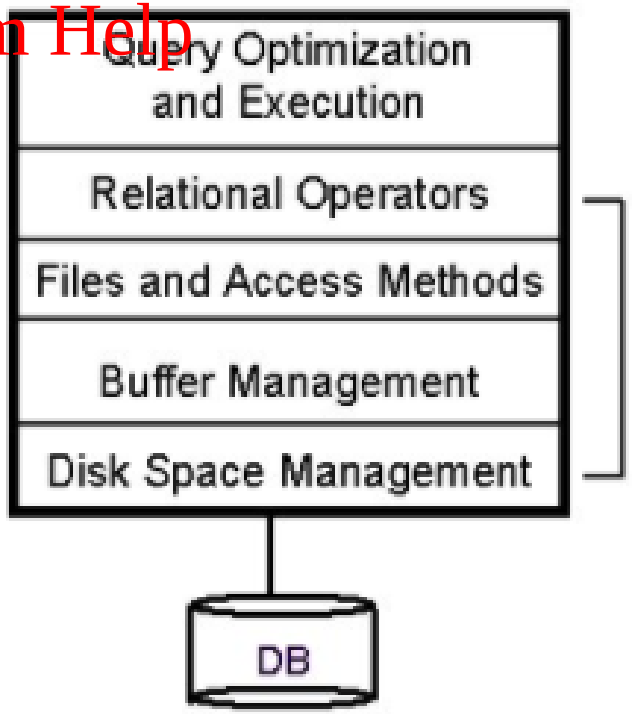
DBMS Architecture – Disk Space Management

- Lowest layer of DBMS software manages space on disk
- Higher levels call upon this layer to:
  - allocate/de-allocate a page
  - read/write a page
- Size of a page = size of a disk block
  - = data unit
- Request for a sequence of pages is often satisfied by allocating contiguous blocks on disk
- Space on disk managed by Disk-space Manager
  - – Higher levels don't need to know how this is done, or how free space is managed



DBMS Architecture – Buffer Management

- Suppose we have 1 million pages in db, but only space for 1000 in memory
- A query needs to scan the entire file
- DBMS has to
  - bring pages into main memory
  - decide which existing pages to replace to make room for a new page
  - called **Replacement Policy**
- Managed by the Buffer manager
  - Files and access methods ask the buffer manager to access a page mentioning the "record id"
  - Buffer manager loads the page if not already there



Disk performance and reliability

- First rule of database performance:
  - Disk access is the most expensive thing databases do

Disk performance and reliability: Problems

- Disks are slow

  => we need a way to improve performance
- Disks are subject to hardware failure

  =>we need a way to protect our data

Disk performance and reliability

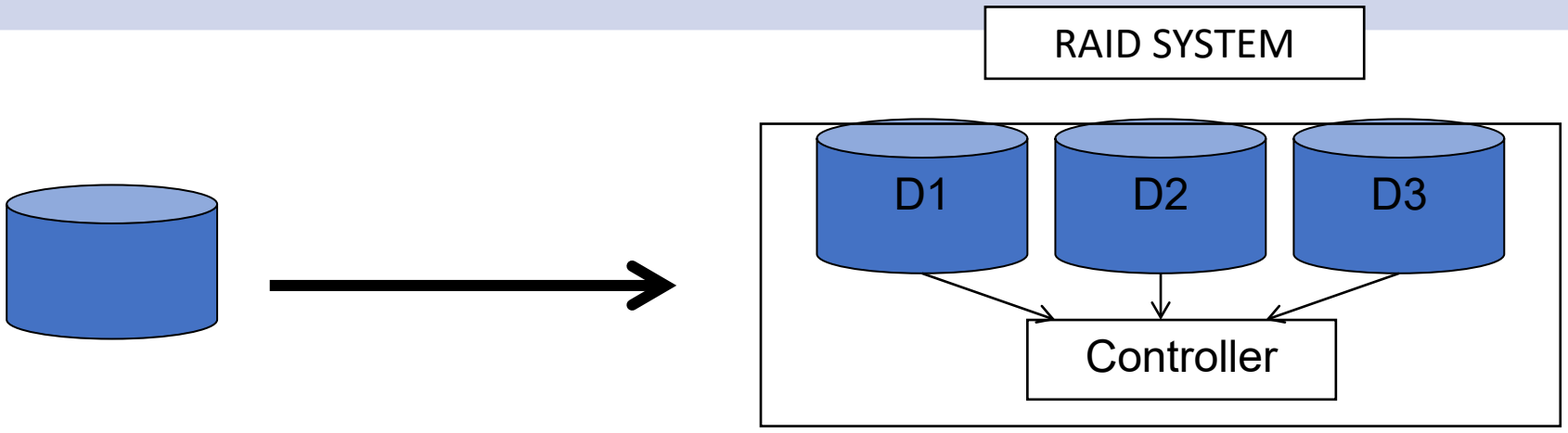
- When using a single disk, we can use scheduling techniques to improve reading/writing performance
- A single HD is often insufficient
  - Limited capacity
  - Limited speed
  - Limited reliability
- If we have multiple disks available
  - We can improve performance spreading data across disks
  - We can improve reliability with redundancy (using spare disks)

Redundant **A**rray of **I**ndependent **D**isks (**RAID**)

Data Virtualization Technology

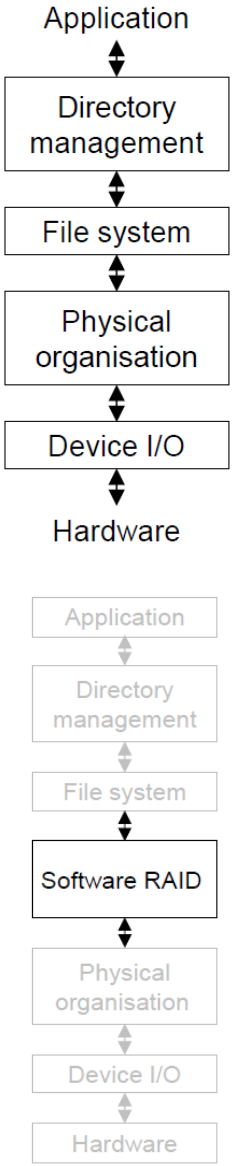
Use more than one physical disk and spread and/or duplicate your data across multiple disks to improve performance and/or reliability (Logically is considered one unit)

Costly since we need to buy and maintain more disks + the hardware needed to make the system working



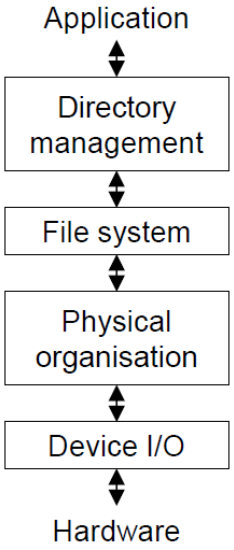
What is RAID?

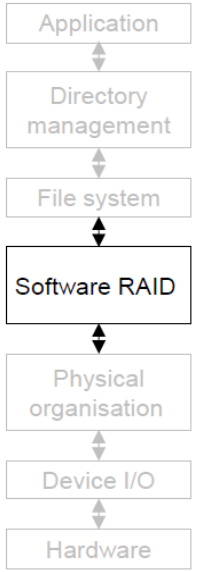
- Raid is a way of organizing file systems over more than one physical disk.
- RAID Array treats multiple hardware disks as a single logical disk
  - More HDs for increased capacity
  - Parallel access for increased speed
  - Controlled redundancy for increased reliability



What is RAID?

- It is usually hardware assisted (RAID controllers, dedicated buses)
- It can also be software only (part of the OS).
  - Windows and Unix have RAID capabilities
- Rationale
  - Disk unit costs is decreasing
  - Dealing with mission critical systems - cost of failure will be larger than not addressing this



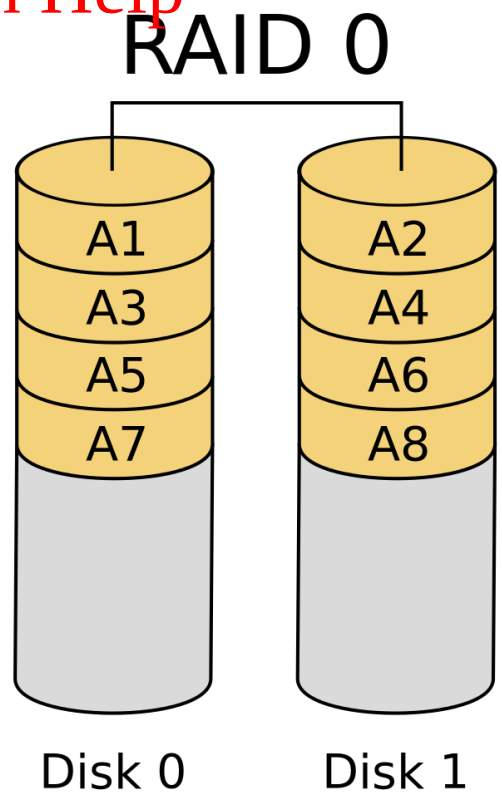


RAID Systems Performance Criteria

- **Speed**: read and write performance
- **Redundancy**: level of reliability
  - If a disk fails, can I recover my data? What if two disks fail?
- **Cost**: how much does the RAID implementation cost?
- **Storage Efficiency**: how much storage is needed to store data?
  - Since data might be replicated or control data might be added to actual data, not all the disk space is used efficiently for data, but only part of it

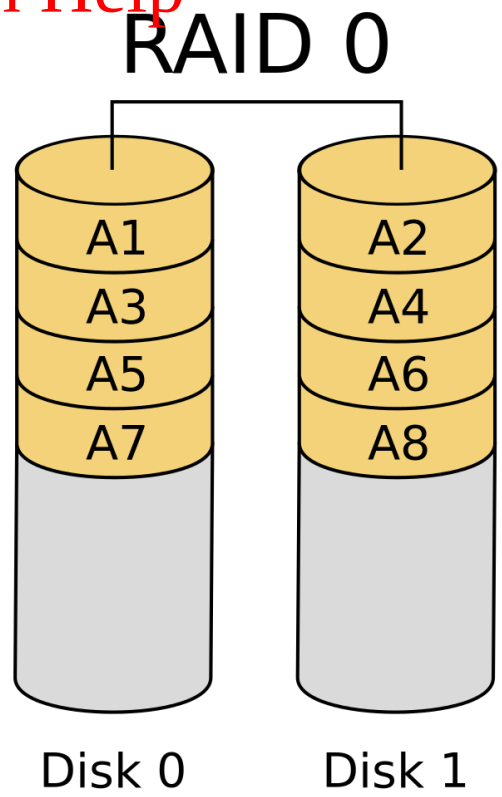
RAID 0 - Striping

- Improve performance by parallelism
- Idea: Distribute data among all disks for increased performance
  - i.e. Two contiguous blocks on separate disks.



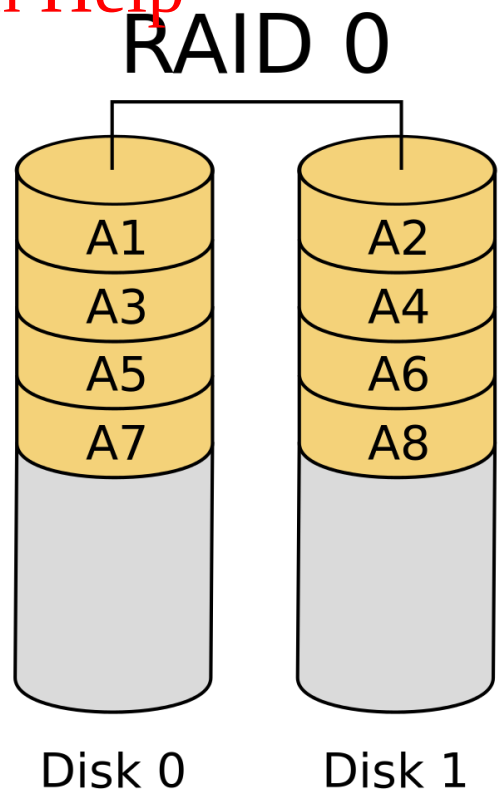
RAID 0 - Striping

- BitLevel Striping:
  - Split all bits of a byte to the disks
  - – e.g. for 8 disks, write i-th bit to disk i
  - Number of disks needs to be a power of 2
  - Each disk is involved in each access
    - Access rate does not increase
    - Read and write transfer speed linearly increases
    - Simultaneous accesses not possible
- Good for speeding up few, sequential and large accesses



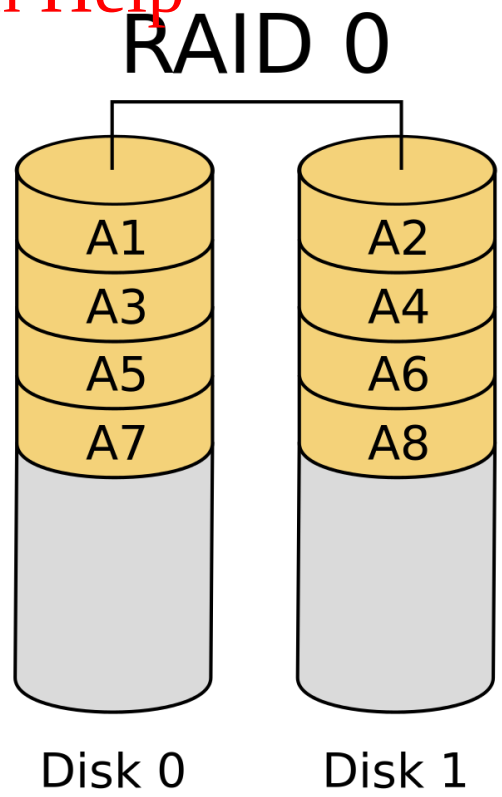
RAID 0 - Striping

- Block Level Striping: Distribute blocks among the disks
  - Only one disk is involved reading a specific block
    - Read and write speed of a single block not increased
    - Other disks still free to read/write other blocks
    - Read and write speed of multiple accesses increase
  - Good for large number of parallel accesses

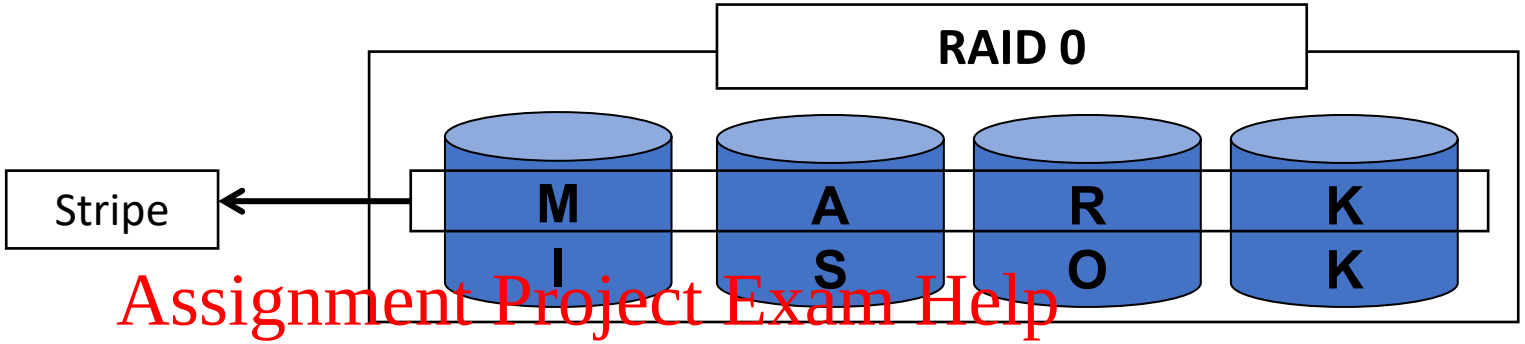


RAID 0 - Striping

- Granularity of stripes (size of a block sector) is very important
  - Fine-grained stripes:
    - Small block size, many disks used for big size file.
  - Coarse-grained strips:
    - Strip bigger.
    - Generally files distributed over less disks. Waste of performance and space for small files.

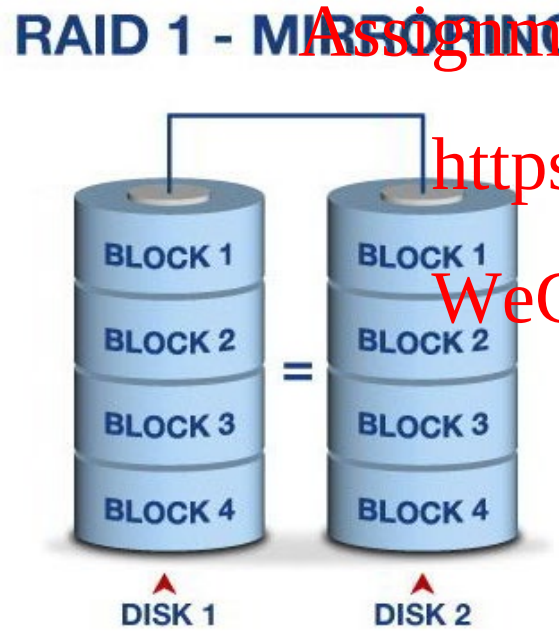


RAID 0 - Striping



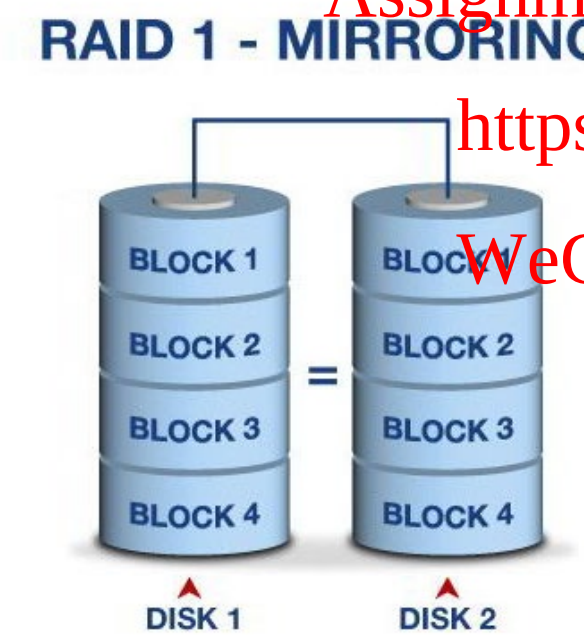
- **Storage Efficiency**:
  - All the space used for data (100% efficiency).
- **Redundancy:**
  - NOT present - No Fault tolerance
- **Performance**:
  - Reading: n times faster (ideally) than single disk
  - Writing: n times faster (ideally).
  - Read and write can overlap
    - E.g.: K can be written while reading A.
  - Very good if sequential
  - The choice of the size of the block is important
- **Cost**:
  - The lowest of all RAID

RAID 1 - Mirroring



- Each data block has a copy on 1 or N disks.
- **Storage Efficiency**: Only 1/N storage space is used. (In case of a mirrored pair, 50% wasted)
- Complete **Redundancy**
  - Fault tolerance is very high.
- Spare disk usually hot swappable (replaceable without stopping DBMS)

RAID 1 - Mirroring



- **High Costs**
- **High Reliability.**
- **Performance:**
  - Writing: slower than average to the same mirror couple (wait the slower one).
  - Reading: improved, (read the fastest disk).
    - Response time decreases by 33%
  - Multiple read, but no multiple write
  - It is not a substitute for a backup strategy

Questions

1. Suppose you have disks of size 20 GB. You have 50 GB of data
   - How many disks (at least) are needed for RAID 0?
   - How many disks (at least) for RAID 1?
   - If data are not critical, which system would you chose?
2. In RAID 1, if 1 mirrored disk fails, can you still keep on working?
3. Why does the size of the block affect performance?

Error Correction Codes

- Increase reliability with computed redundancy
- The **parity bit** is an extra-bit added to a sequence of data used to check for data errors

Error Correction Codes

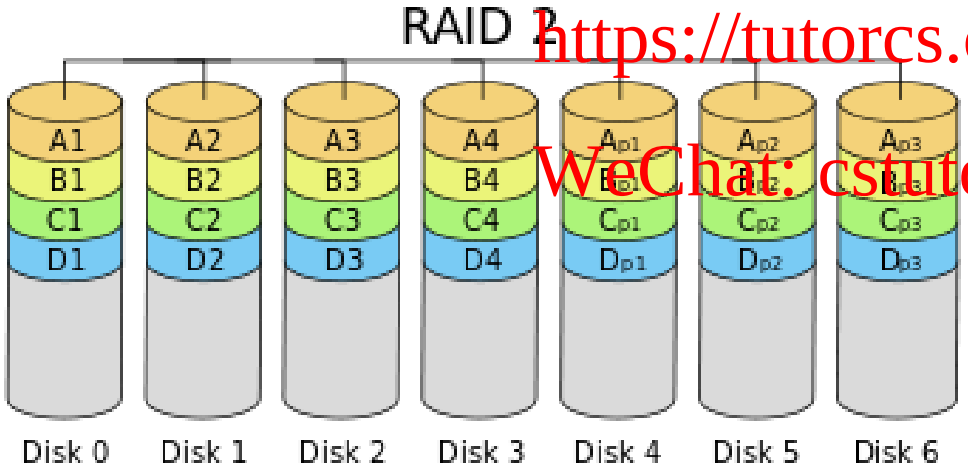
- Suppose your data are a sequence of 5 bits:

Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Parity
0	1	1	0	1	??

- The parity bit has a value so that your data + the parity bit has an **even** number of ones:

Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	**Parity**
0	1	1	0	1	**1**

RAID 2

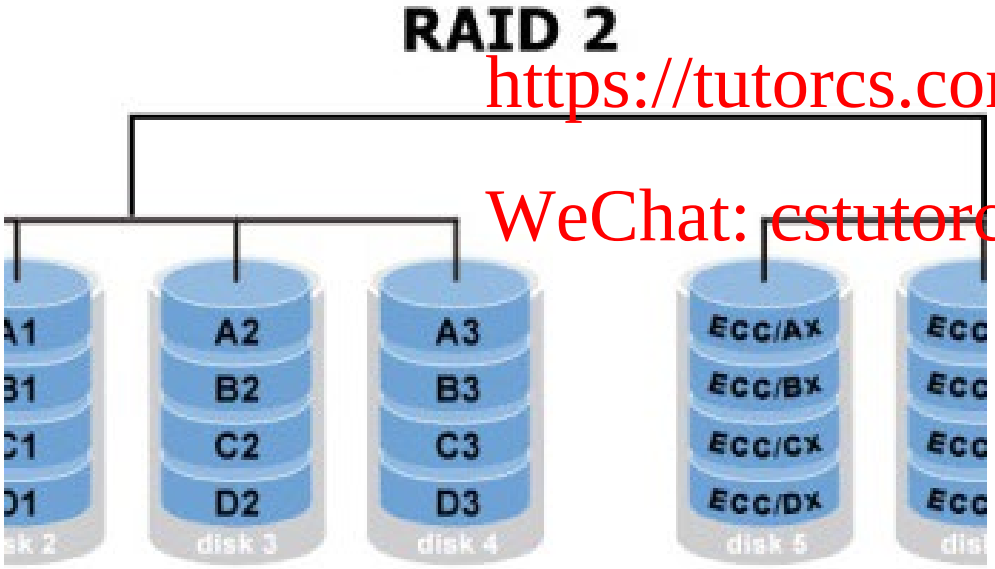


- Uses bit-level striping and each sequential bit is placed on a different hard drive.
- The error correcting code (ECC) used is the Hamming code parity, which is calculated across bits and stored separately in at least a single drive.

Parity Bit – why it is used?

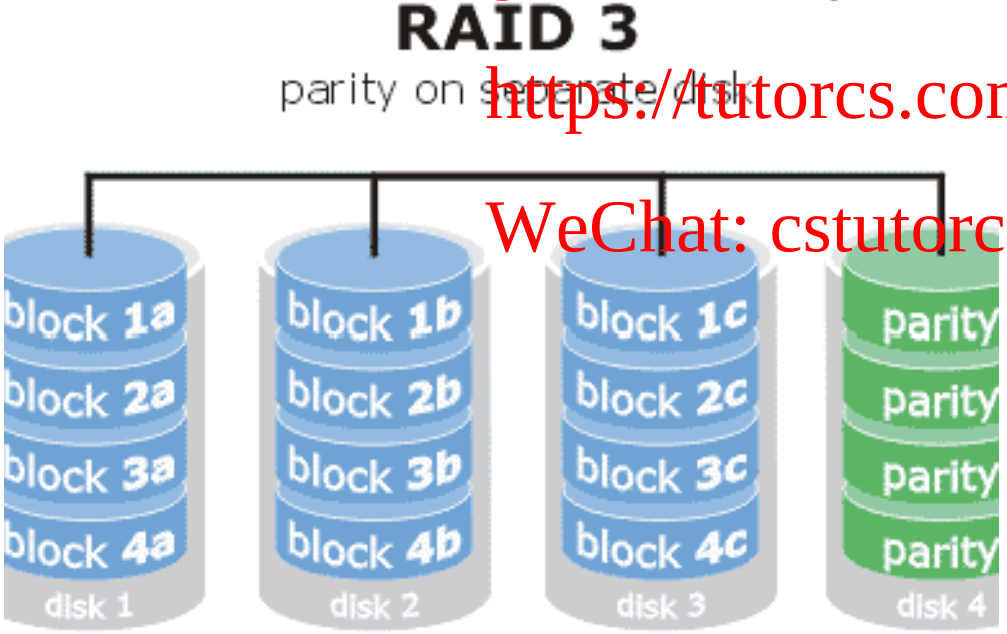
- Detect errors in your data
- What if there are two errors?
  - More parity bits are needed to detect them.
  - E.g. the Hamming Code (7,4) adds 3 parity bits every 4 bits of data to detect and correct errors
- Why are they used with RAID?
  - If we store parity bits on a disk, the parity bit can be used to reconstruct a faulty disk in case of failure
- Not really used in industry anymore

RAID 2



- Rarely used
- **Storage efficiency**: low.
- **Fault Tolerance**: 2 disks can break.

RAID 3

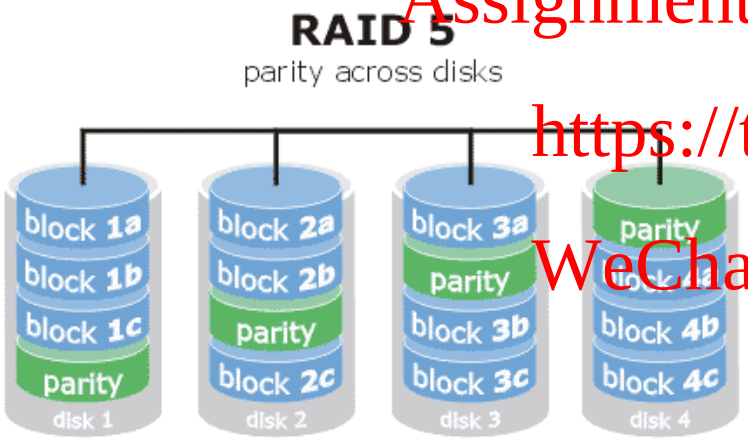


- Rarely used in practice
- Byte level striping
- A dedicated parity disk is added
- The other disks are striped (RAID 0)

RAID 3 – The bottleneck problem

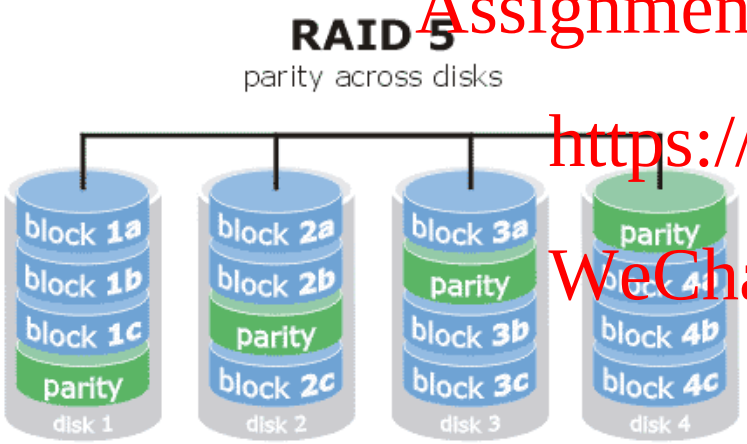
- **Storage efficiency:** only 1 disk used for control. So, (N-1)/N. with N=10 storage efficiency is 90%
- **Fault Tolerance:** one disk failure
- **Cost**: Fair. Hardware controller required
- **Performance**
  - Writing: quite poor.
    - Parity bits must be computed for every stripe and written. Even if part of the stripe is written (in that case all the stripe must be read before writing)
  - The parity disk limits performance and represents a bottleneck

RAID 5



- The most popular solution.
- Parity blocks are striped as well.
- Removes the bottleneck on the parity disk for RAID 3

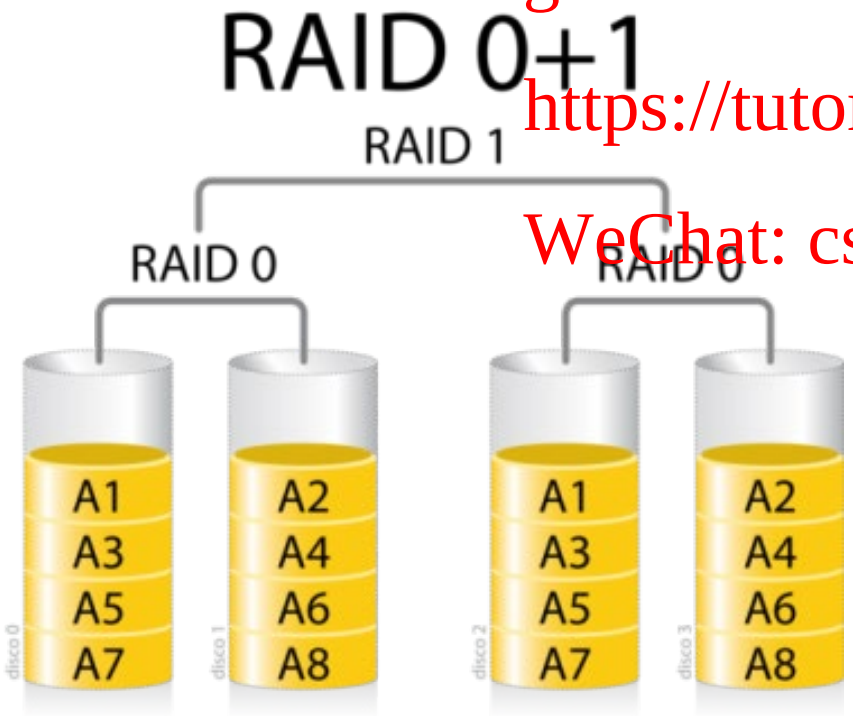
RAID 5

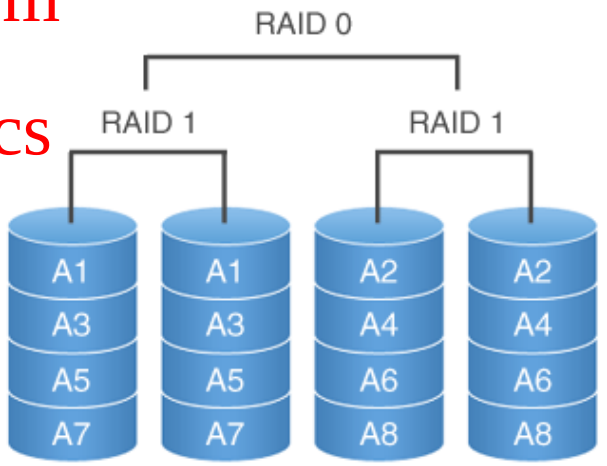


- **Storage efficiency.** like RAID 3.
- **Fault Tolerance** like RAID 3
- **Performance**.
  - Writing is improved because parity disks are different.
    - Again, all the stripe must be read to compute parity.
  - Improvement using AFRAID techniques (parity computed every X ms instead of every time)
- **Cost**: fair

RAID 10 (RAID 0+1)

- Combination of RAID 0 (for performance) and RAID 1 (for reliability)
- RAID 10: first RAID 1 then RAID 0
- RAID 0+1: first RAID 0 then RAID 1
- Mirroring duplicates all your data.
- Fast because the data is striped across multiple disks
- Chunks of data can be read and written to different disks simultaneously.



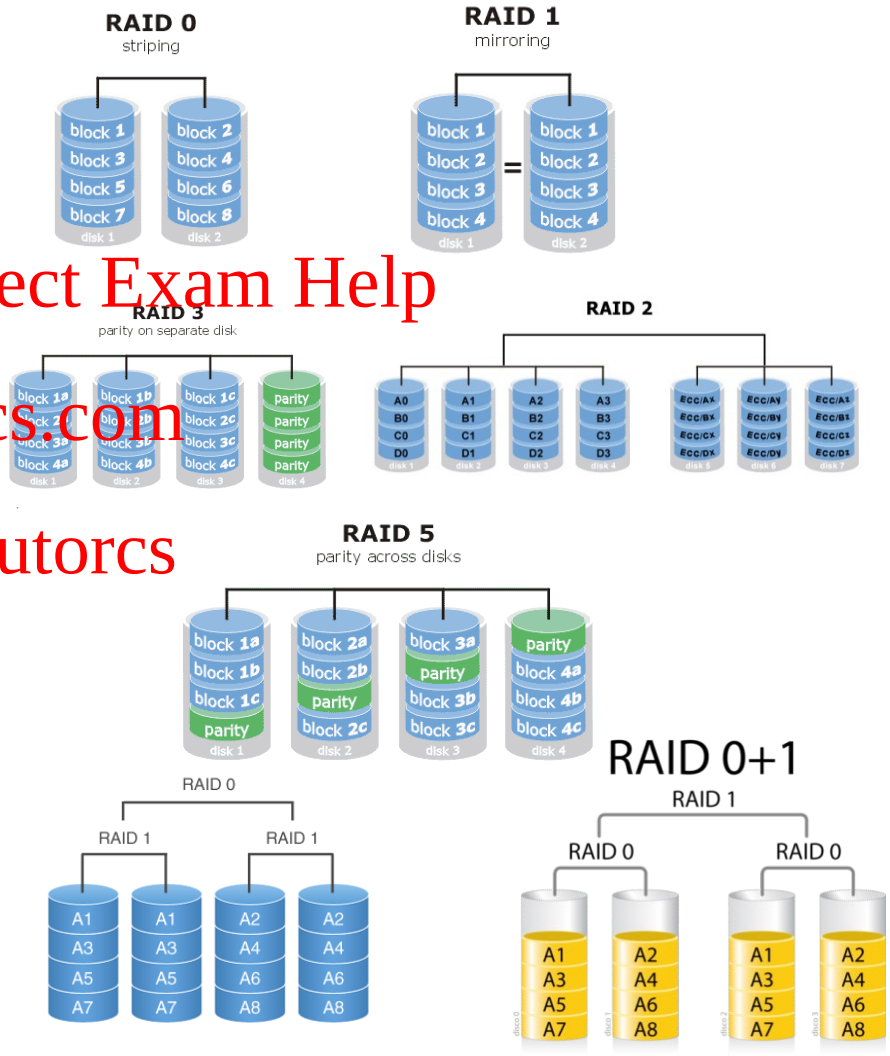


RAID 10 and RAID 0+1

- Low Storage Efficiency (50%)
- **High performance** due to striping
- **High reliability** due to mirroring
- **High Cost**
- RAID 10 is the Industry Standard

- RAID is implemented by Storage servers containing arrays of disks connected by optical fibres and connected to a server farm via a dedicated private storage area network (SAN)

RAID	
0	Striping. Data spread across disks
1	Mirroring. Each disk has one or more identical copy
2	Multiple parity disks to detect and correct more errors
3	Parity disk added
5	Parity bit spread across multiple disks
10	Disk are mirrored, then striped (stripe of a mirror)
01	Disks are striped, then mirrored (mirror of a stripe)



Comparison

- RAID 10 slightly better than RAID 0+1

RAID LEVEL	Storage Efficiency	Performance	Write Concurrency	Read Concurrency	Redundancy	Costs
0	100%	High	Y	Y	None	Low
1	50% (or lower)	Low	N	Y	Mirroring	High
3	Medium	Medium	N	Y (N)	Parity	Fair
5	Medium	Medium	N	Y (N)	Parity	Fair
10	50% (low)	High	Y	Y	Mirroring	High
01	50% (low)	High	Y	Y	Mirroring	High

Conclusions

- **RAID systems improve performance and reliability**
- Which one is the best choice?
  - It depends on performance required, budget and criticality of data
  - RAID 10 is the most used standard in industry for big companies with large budget and medium dataset
  - RAID 5 is popular (cheaper solution but still good performance)
  - Critical data – such as OS – are usually mirrored
  - Non critical data can be striped only
  - Non critical data with little access do not require RAID