DBMS Tutorial 07.11.2018

Disk Scheduling Algorithms

- First Come First Serve
- Elevator

RAID

Disk Scheduling - FCFS

- First Come First Serve
  - Process requests sequentially
  - Fair to all processes
  - Approaches random scheduling in performance if there are many processes

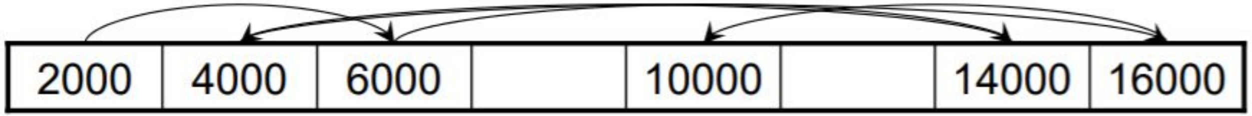
Disk Scheduling - Elevator

- Elevator Algorithm(Also called SCAN sometimes)
  - Arm moves in one direction only, satisfying all outstanding requests until it reaches the last track in that direction
  - Direction is reversed if there are no outstanding requests in that direction (or if its on the last track in that direction)

First-Come-First-Serve Scheduling

	2000	2000
+	|6000-2000|	4000
+	|14000-6000|	8000
+	|4000-14000|	10000
+	|16000-4000|	12000
+	|10000-16000|	6000
=		42000

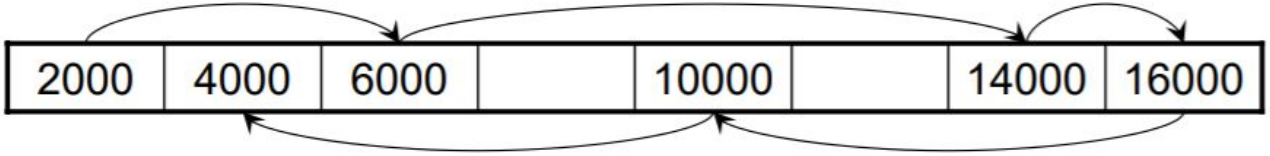
Cylinder of request	First time available	Time completed
2000	0	4.42
6000	0	13.84
14000	0	27.26
4000	10	42.68
16000	20	60.10
10000	30	71.52



Elevator Algorithm

	2000	2000
+	|6000-2000|	4000
+	|14000-6000|	8000
+	|16000-14000|	2000
+	|10000-16000|	6000
+	|4000-10000|	6000
=		28000

Cylinder of request	First time available	Time completed
2000	0	4.42
6000	0	13.84
14000	0	27.26
4000	10	57.52
16000	20	34.68
10000	30	46.10

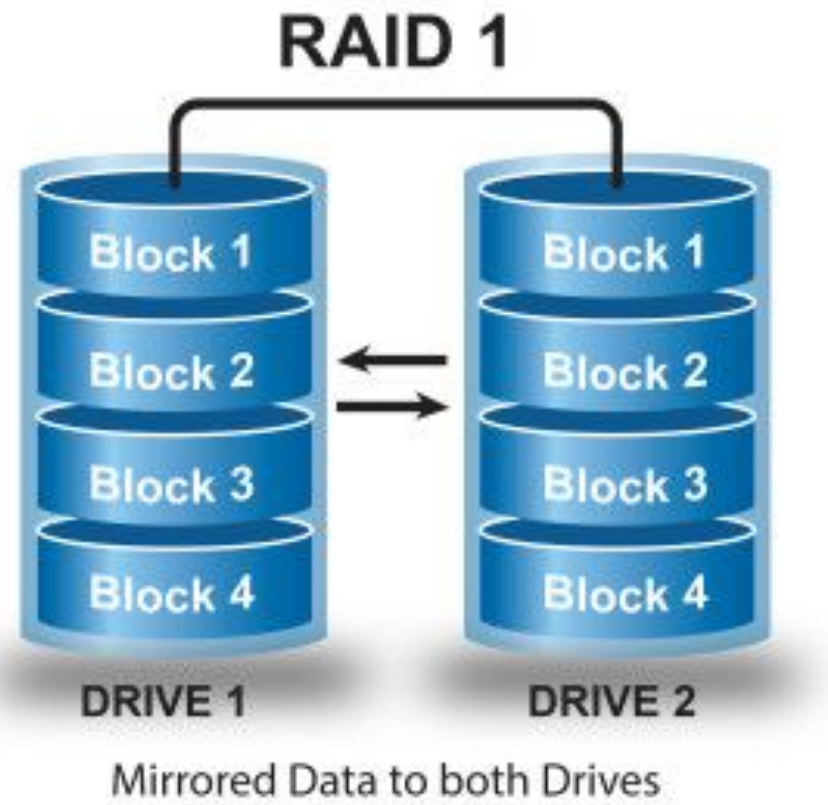


RAID

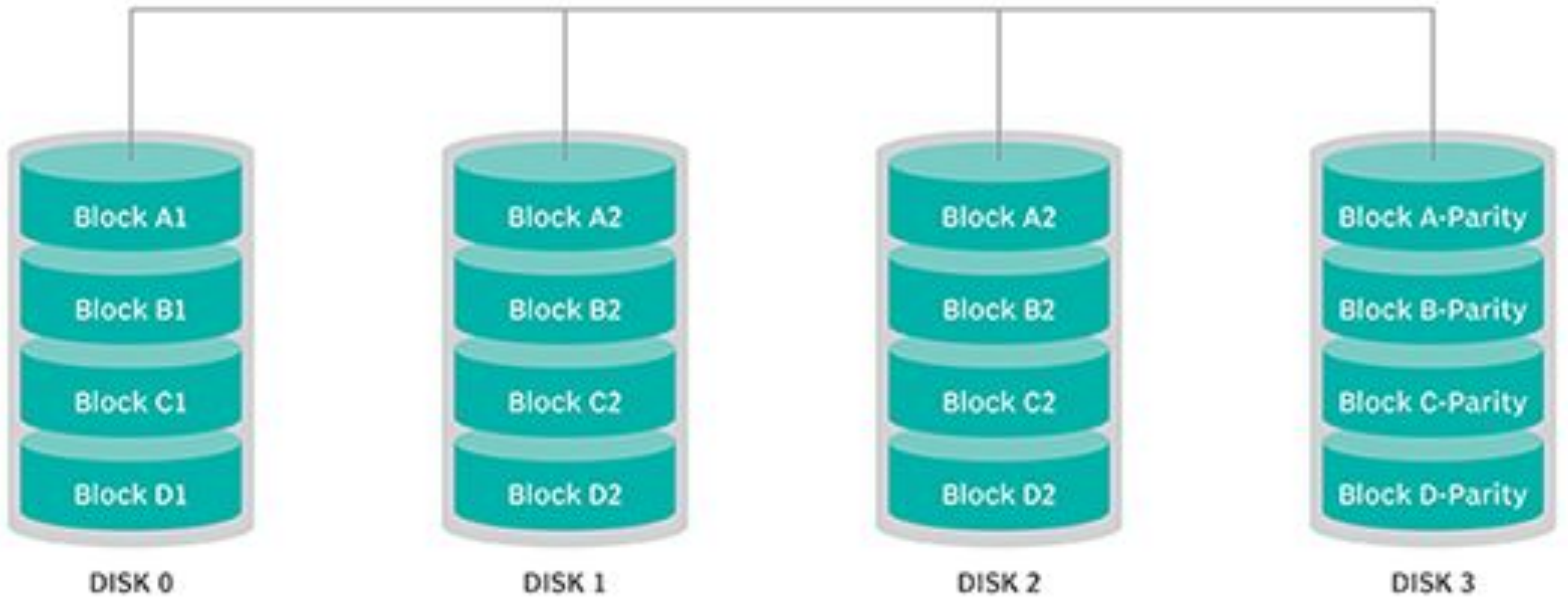
- A RAID is a Redundant Array of Inexpensive/Indepedent Disks
- The alternative is SLED, single large expensive disk
- Disks are small and cheap, so it’s easy to put lots of disks (10s to 100s) in one box for increased storage, performance, and availability
- The RAID box with a RAID controller looks just like a SLED to the computer
- **Data plus some redundant information is striped across the disks in some way**
- **How that striping is done is key to performance and reliability.**

RAID 1 (mirroring)

- Mirrored Disks
- Data is written to two places
- On failure, just use surviving disk
- On read, choose fastest to read
- Write performance is same as
- single drive (but need to write in two places),
- read performance can be 2x better
- Replication redundancy is expensive



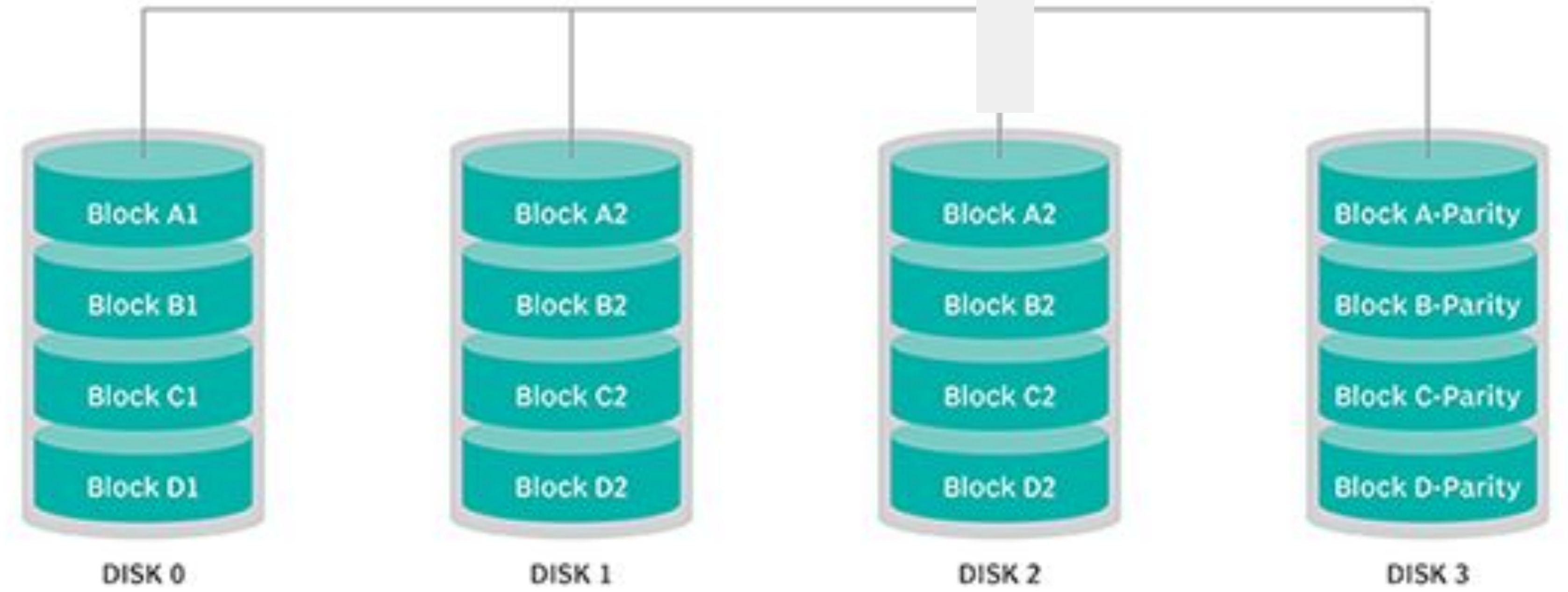
RAID 4



Use n data disks and one redundant disk.Each block of the redundant disk is the parity check for the corresponding blocks of the data disks.

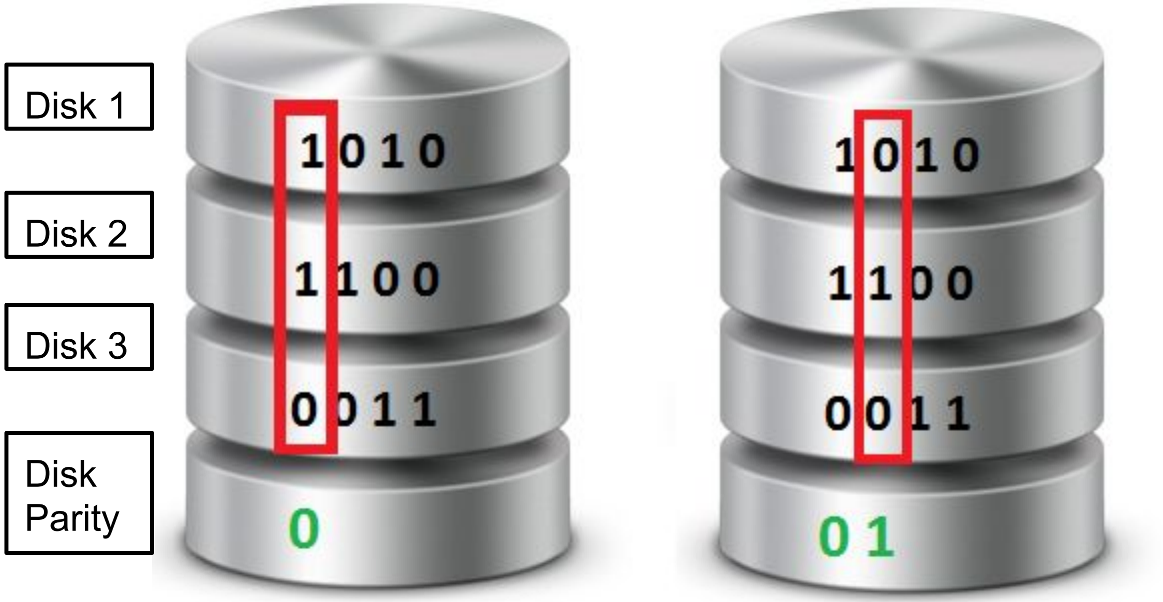
RAID 4

Please consider these blocks to be named A3, B3, C3 and D3.



$$A1 \oplus A2 \oplus A3 = \text{A-Parity}$$

Calculating the parity bit



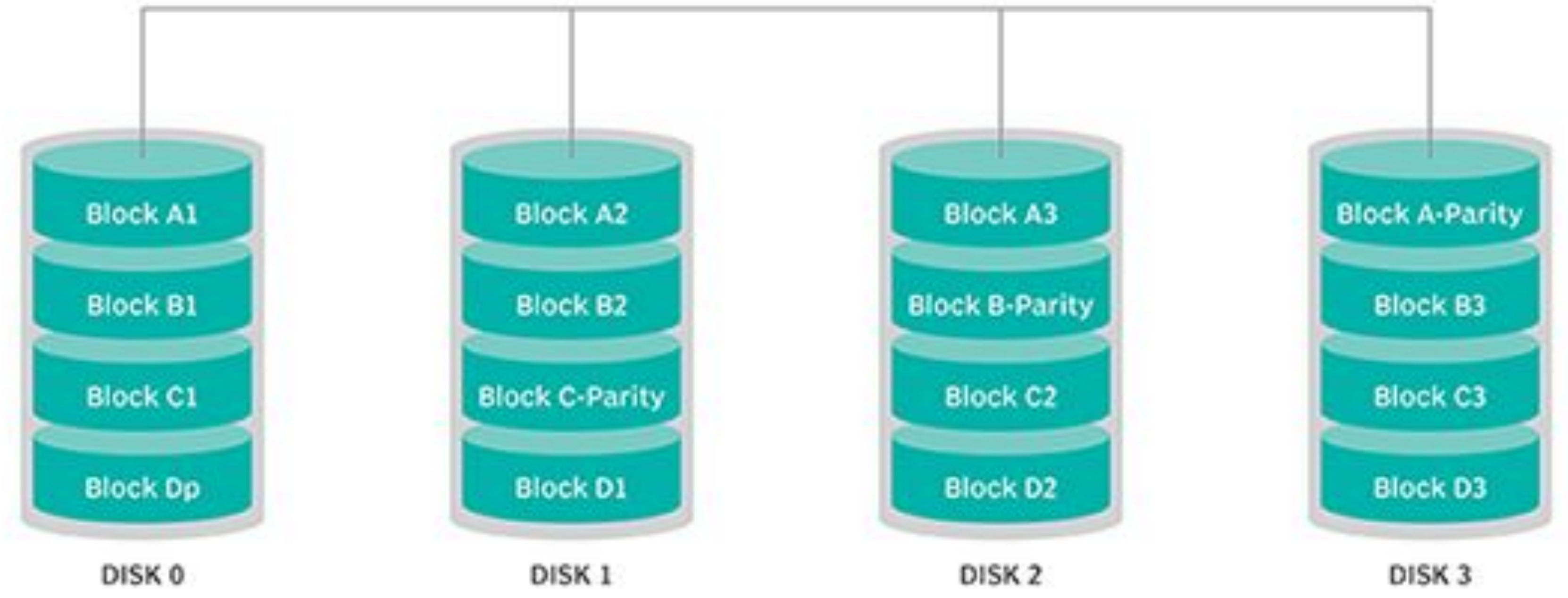
RAID 4

RAID 4 consists of block-level striping with a dedicated parity disk.

- Provides good performance of random reads, while the performance of random writes is low due to the need to write all parity data to a single disk (bottleneck).

In the diagram above, a read request for block A1 would be serviced by disk 0. A simultaneous read request for block B1 would have to wait, but a read request for B2 could be serviced concurrently by disk 1.

RAID 5



$$A1 \oplus A2 \oplus A3 = A\text{-}Parity$$

RAID 5

RAID 5 consists of block-level striping with distributed parity.

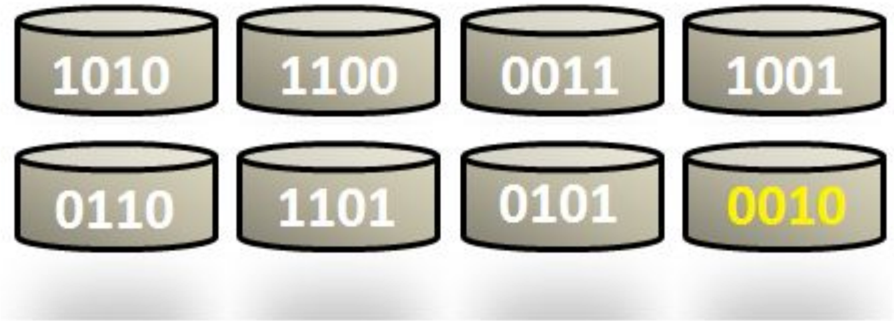
- parity information is distributed among the drives.
- requires at least three disks (two for data and one for parity).
- In comparison to RAID 4, RAID 5's distributed parity evens out the stress of a dedicated parity disk among all RAID members.
- Additionally, write performance is increased since all RAID members participate in the serving of write requests, and therefore this is no longer a bottleneck.

Writing a new block

What must be done?

1. Read the old data
2. Read the old parity
3. Write the new data
4. Write the new parity

Example



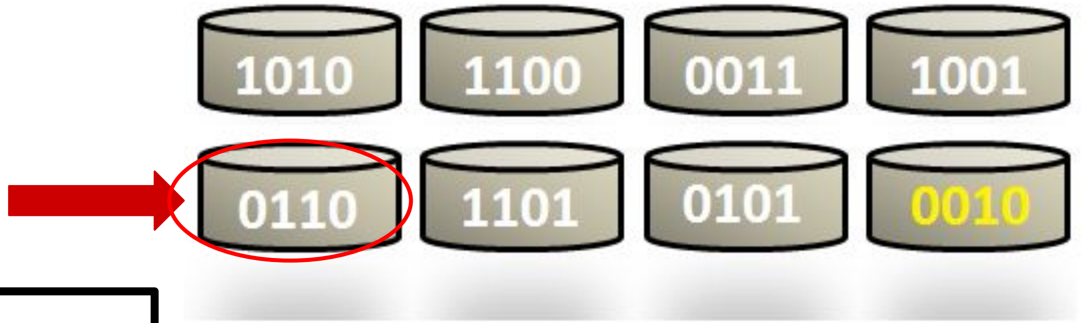
Seven data disks and one parity disk (yellow one).
Disk 5 needs to be changed to new value of 1111.

Writing a new block

What must be done?

1. Read the old data
2. Read the old parity
3. Write the new data
4. Write the new parity

Example



Old parity, **z** = 0010
Old value of Disk 5, **y** = 0110
New Value of Disk 5, **x** = 1111

New Parity = x ⊕ y ⊕ z = 1011

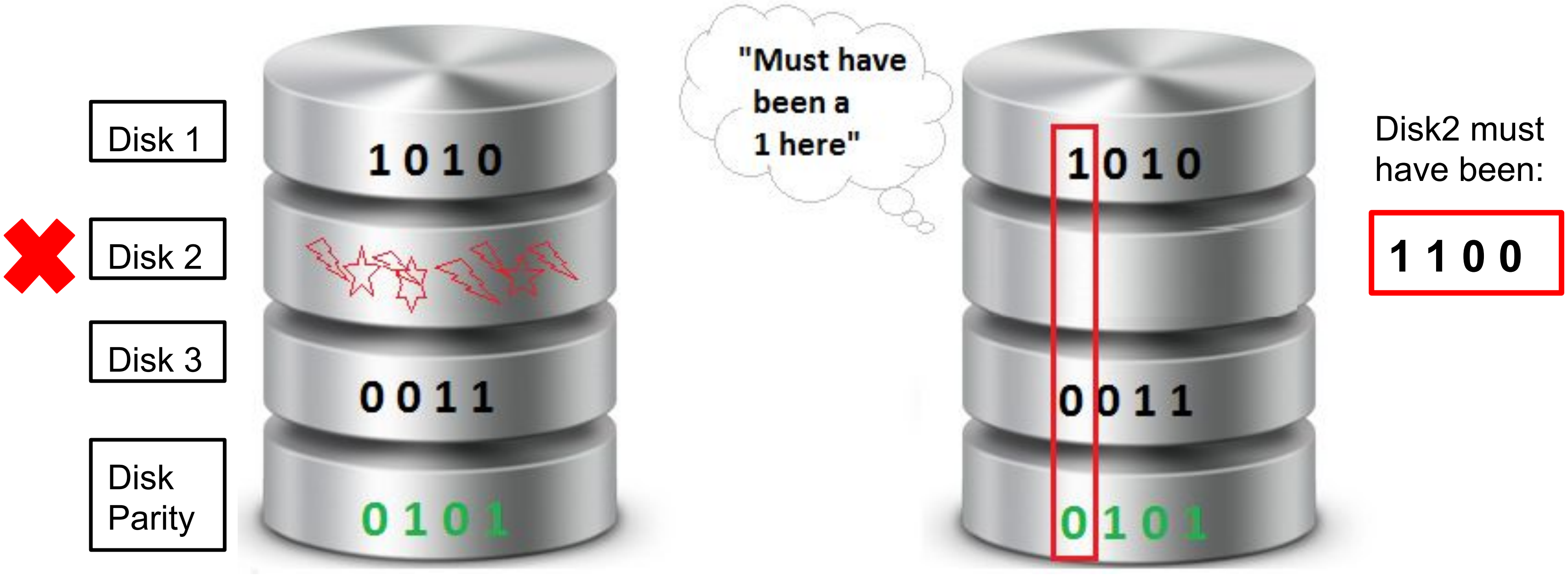
Seven data disks and one parity disk (yellow one).
Disk 5 needs to be changed to new value of 1111.

Error Recovery

$$\textbf{Disk1} \oplus \textbf{Disk2} \oplus \textbf{Disk3} = \textbf{Disk}_{\textbf{parity}}$$

$$\textbf{Disk2} = \textbf{Disk1} \oplus \textbf{Disk3} \oplus \textbf{Disk}_{\textbf{parity}}$$

Error Recovery



$$\textbf{Disk1} \oplus \textbf{Disk2} \oplus \textbf{Disk3} = \textbf{Disk}_{\textbf{parity}}$$

$$\textbf{Disk2} = \textbf{Disk1} \oplus \textbf{Disk3} \oplus \textbf{Disk}_{\textbf{parity}}$$

Error Recovery

$$\mathbf{Disk1 \oplus Disk2 \oplus Disk3 = Disk_{parity}}$$

$$\mathbf{Disk2 = Disk1 \oplus Disk3 \oplus Disk_{parity}}$$

NOTE :
Parity bit is chosen such that
$\text{Disk1} \oplus \text{Disk2} \oplus \text{Disk3} \oplus \text{Disk}_{parity} = 0$

Useful links

https://datapacket.com/blog/advantages-disadvantages-various-raid-levels/

http://rickardnobel.se/how-raid5-works/

http://dbmsfortech.blogspot.com/2016/05/levels-of-raid-level-of-redundancy.html

https://en.wikipedia.org/wiki/Standard_RAID_levels