Database Security and Administration

Part III: Backup and Recovery

INT206 Advanced Database

Sanit Sirisawatvatana and Sunisa Sathapornvajana

Lab 5 Discussion

- Issues found in Lab 5.
  - Role does NOT effect current sessions.
  - Activate roles between using SET ROLE command and using SET DEFAULT ROLE command.
  - MySQL: The SELECT, INSERT, UPDATE and DELETE privileges on the table can be granted at the column level.

Database Security

- **Goal of database security**
  - To minimize losses caused by predictable events in a cost-effective manner without excessively disturbing the user
- **Computer-based Security Controls**
  - Authorization
  - Access controls
  - Views
  - Backup and recovery
  - Integrity
  - Encryption
  - RAID technology

Database Security

- To protect data from the following situations:
  - Theft and fraud
  - Loss of confidentiality (maintain secrecy)
  - Loss of privacy (protect individual data)
  - **Loss of integrity** (invalid or corrupted data)
  - **Loss of availability** (access data or system)

Integrity

- Integrity constraints
  - Contribute to maintaining a secure database system by providing data from becoming invalid, and hence giving misleading or incorrect results.
- Types of integrity constraints
  - Required data
  - Domain constraints
  - Multiplicity
  - Entity integrity
  - Referential integrity
  - General constraints

Errors and Failures Requiring Recovery from Backup

- **Human Errors**
  - Human errors occur when, either due to an error in application logic or a manual mis-step, data in your database is changed or deleted incorrectly. Data loss due to human error includes such missteps as dropping important tables or deleting or changing the contents of a table. Human error is the number one cause of data loss.
- **Media Failures/Disk Failure**
  - Due to physical damage or a logical failure. Disk failure is probably one of the most common causes of data loss.

Backup and Recovery

- Backup
  - Process of periodically taking a copy of the database and log file (and possibly programs) to offline storage media (disk or tape).

- Journaling
  - Process of keeping and maintaining a log file (or journal) of all changes made to database to enable effective recovery in event of failure.

- DBMS should provide backup and recovery facilities to assist with the recovery of a database failure.

- With the journaling, the database can be recovered to its last known consistent state using a backup copy and the information contain in the log file.

Backup Concepts

1. What data need to be backed up?

2. Location of systems and files to be backed up?

3. Who performs backups?

4. Time frames for backups?

Backup Concepts in MySQL

It is important to back up your databases so that you can recover your data and be up and running again in case problems occur, such as system crashes, hardware failures, or users deleting data by mistake. Backups are also essential as a safeguard before upgrading a MySQL installation, and they can be used to transfer a MySQL installation to another system or to set up replica servers.

MySQL offers a variety of backup strategies from which you can choose the methods that best suit the requirements for your installation. This chapter discusses several backup and recovery topics with which you should be familiar:

- Types of backups: Logical versus physical, full versus incremental, and so forth.
- Methods for creating backups.
- Recovery methods, including point-in-time recovery.
- Backup scheduling, compression, and encryption.
- Table maintenance, to enable recovery of corrupt tables.

Backup and Recovery Types

Physical (Raw) vs Logical Backups

Physical backups consist of raw copies of the directories and files that store database contents. This type of backup is suitable for large, important databases that need to be recovered quickly when problems occur.

Logical backups save information represented as logical database structure (CREATE DATABASE, CREATE TABLE statements) and content (INSERT statements or delimited-text files). This type of backup is suitable for smaller amounts of data where you might edit the data values or table structure, or recreate the data on a different machine architecture.

Physical Backups

Physical backup methods have these characteristics:

- The backup consists of exact copies of database directories and files. Typically this is a copy of all or part of the MySQL data directory.
- Physical backup methods are **faster** than logical because they involve only file copying without conversion.
- Backup and restore granularity ranges from the level of the entire data directory down to the level of individual files. This may or may not provide for table-level granularity, depending on storage engine. For example, InnoDB tables can each be in a separate file, or share file storage with other InnoDB tables; each MyISAM table corresponds uniquely to a set of files.
- In addition to databases, the backup can include any related files such as log or configuration files.
- ***Backups are portable only to other machines that have identical or similar hardware characteristics.***

Physical Backups

Physical backup methods have these characteristics:

- Backups can be performed while the MySQL server is not running. If the server is running, it is necessary to perform appropriate locking so that the server does not change database contents during the backup. **MySQL Enterprise Backup does this locking automatically for tables that require it.**
- Physical backup tools include the **mysqlbackup** of MySQL Enterprise Backup for InnoDB or any other tables, or file system-level commands (such as **cp**, **scp**, **tar**, **rsync**) for MyISAM tables.
- For restore:
  - MySQL Enterprise Backup restores InnoDB and other tables that it backed up.
  - **ndb_restore** restores NDB tables.
  - Files copied at the file system level can be copied back to their original locations with file system commands.

https://dev.mysql.com/doc/mysql-enterprise-backup/8.0/en/mysqlbackup.html

Logical Backups

Logical backup methods have these characteristics:

- The backup is done by querying the MySQL server to obtain database structure and content information.
- Backup is **slower** than physical methods because the server must access database information and convert it to logical format. If the output is written on the client side, the server must also send it to the backup program.
- **Output is larger** than for physical backup, particularly when saved in text format.
- Backup and restore granularity is available at the server level (all databases), database level (all tables in a particular database), or table level. This is true regardless of storage engine.
- The backup does not include log or configuration files, or other database-related files that are not part of databases.
- **Backups stored in logical format are machine independent and highly portable.**
- Logical backups are performed with the MySQL server running. The server is not taken offline.
- Logical backup tools include the **mysqldump** program and the SELECT ... INTO OUTFILE statement. These work for any storage engine, even MEMORY.
- **To restore logical backups**, SQL-format dump files can be processed using the **mysql** client. **To load delimited-text files**, use the LOAD DATA statement or the **mysqlimport** client.

Online vs Offline Backups

Online backups take place while the MySQL server is running so that the database information can be obtained from the server. Offline backups take place while the server is stopped. This distinction can also be described as “hot” versus “cold” backups; a “warm” backup is one where the server remains running but locked against modifying data while you access database files externally.

Online backup methods have these characteristics:

- The backup is less intrusive to other clients, which can connect to the MySQL server during the backup and may be able to access data depending on what operations they need to perform.
- Care must be taken to impose appropriate locking so that data modifications do not take place that would compromise backup integrity. The MySQL Enterprise Backup product does such locking automatically.

Online vs Offline Backups

Offline backup methods have these characteristics:

- Clients can be affected adversely because the server is unavailable during backup. For that reason, such backups are often taken from a replica that can be taken offline without harming availability.
- The backup procedure is simpler because there is no possibility of interference from client activity.

A similar distinction between online and offline applies for recovery operations, and similar characteristics apply. However, it is more likely for clients to be affected by online recovery than by online backup because recovery requires stronger locking. During backup, clients might be able to read data while it is being backed up. Recovery modifies data and does not just read it, so clients must be prevented from accessing data while it is being restored.

Full vs Incremental Backups

A **full backup** includes all data managed by a MySQL server at a given point in time.

An **incremental backup** consists of the changes made to the data during a given time span (from one point in time to another).

MySQL has different ways to perform full backups, such as those described earlier in this section. Incremental backups are made possible by enabling the server's binary log (use the mysqlbinlog utility), which the server uses to record data changes.

Full vs Point-in-Time (Incremental) Recovery

A full recovery restores all data from a full backup. This restores the server instance to the state that it had when the backup was made. If that state is not sufficiently current, a full recovery can be followed by recovery of incremental backups made since the full backup, to bring the server to a more up-to-date state.

Incremental recovery is recovery of changes made during a given time span. This is also called point-in-time recovery because it makes a server's state current up to a given time. **Point-in-time recovery is based on the binary log and typically follows a full recovery from the backup files that restores the server to its state when the backup was made.** Then the data changes written in the binary log files are applied as incremental recovery to redo data modifications and bring the server up to the desired point in time.

Backup Concepts in MySQL

It is important to back up your databases so that you can recover your data and be up and running again in case problems occur, such as system crashes, hardware failures, or users deleting data by mistake. Backups are also essential as a safeguard before upgrading a MySQL installation, and they can be used to transfer a MySQL installation to another system or to set up replica servers.

MySQL offers a variety of backup strategies from which you can choose the methods that best suit the requirements for your installation. This chapter discusses several backup and recovery topics with which you should be familiar:

- Types of backups: Logical versus physical, full versus incremental, and so forth.
- Methods for creating backups.
- Recovery methods, including point-in-time recovery.
- Backup scheduling, compression, and encryption.
- Table maintenance, to enable recovery of corrupt tables.

- **Possible backup methods** (Hint: Backup methods depend on the DBMS)
  - **End users**: Backup the tables as a SQL script including CREATE and INSERT statements.
  - **DBA**: Backup the tables using EXPORT utility.
  - **DBA**: Backup the database using Recovery Manager with BACKUP command.

Recovery Concepts

- **In case of dropping important tables:**

- **Possible recovery methods:**
  (Hint: Recovery methods depend on the DBMS)

- **End users**: Recovery the tables by running the SQL script for creating tables and data insertion.

- **End users**: Using Flashback technology to recover the dropped tables (Only in an Oracle DBMS)

- **DBA**: Recovery the tables using IMPORT utility.

- **DBA**: Recovery the database point in time using Backup files and Log files

Recovery Concepts

- **In case of disk failures:**
  - **DBA**: Perform automatic a full database backup schedule.
  - **DBA**: Recovery the database point in time using Backup files and Log files
- **System Crash**:
  - *Automatically* recovered by the DBMS using REDO-type log entries, UNDO-type log entries and Checkpoint process.

Redundant Array of Independent Disk (RAID)

- Fault-tolerant
  - The system should continue to operate even if one of the hardware components fails
  - Requires to have redundant components

- Main hardware components should be fault-tolerant
  - Disk drives
  - Disk controllers
  - CPU
  - Power suppliers
  - Cooling fans

- Disk drives are the most vulnerable components with the shortest times between failure of any of the hardware components.

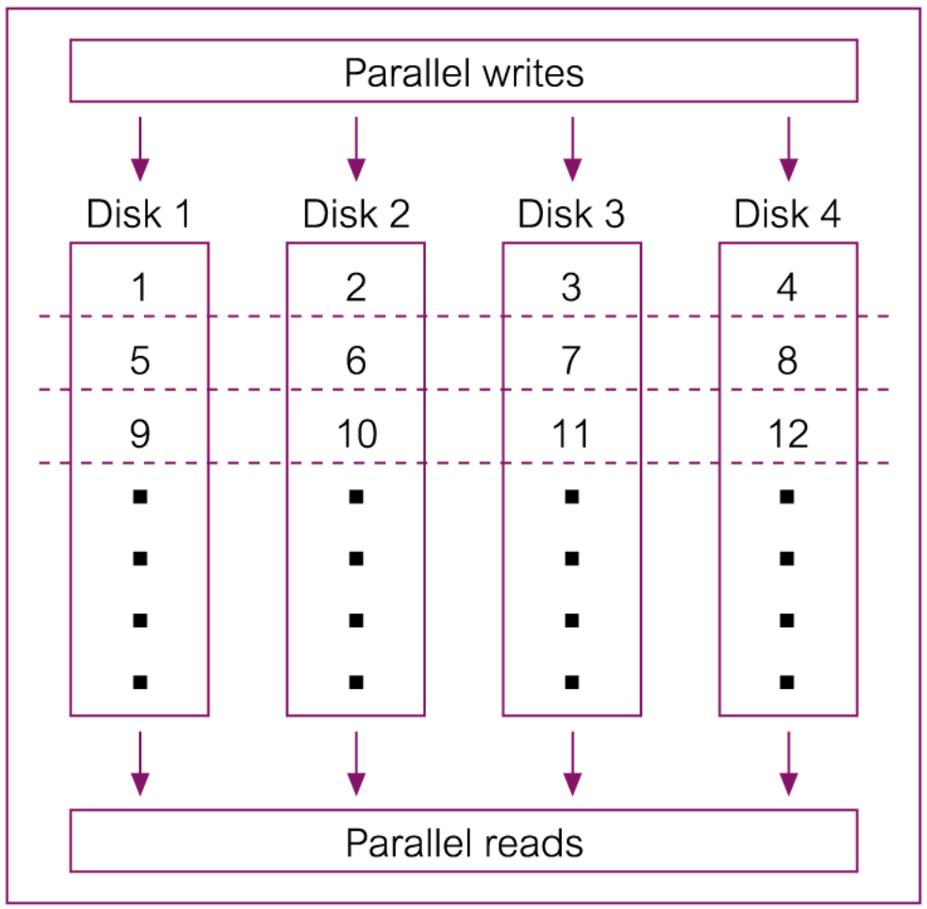
RAID

- Provides a large disk array that operates as a single disk
- Consists of an arrangement of several independent disks
- Are organized to improve reliability and at the same time increase performance.
- Performance increasing through data striping
  - The data is segmented into equal-size partitions (the striping unit), which are transparently distributed across multiple disks.
- Reliability Improvement through data redundant
  - The data is storing information across the disks using a parity scheme or an error-correcting scheme.

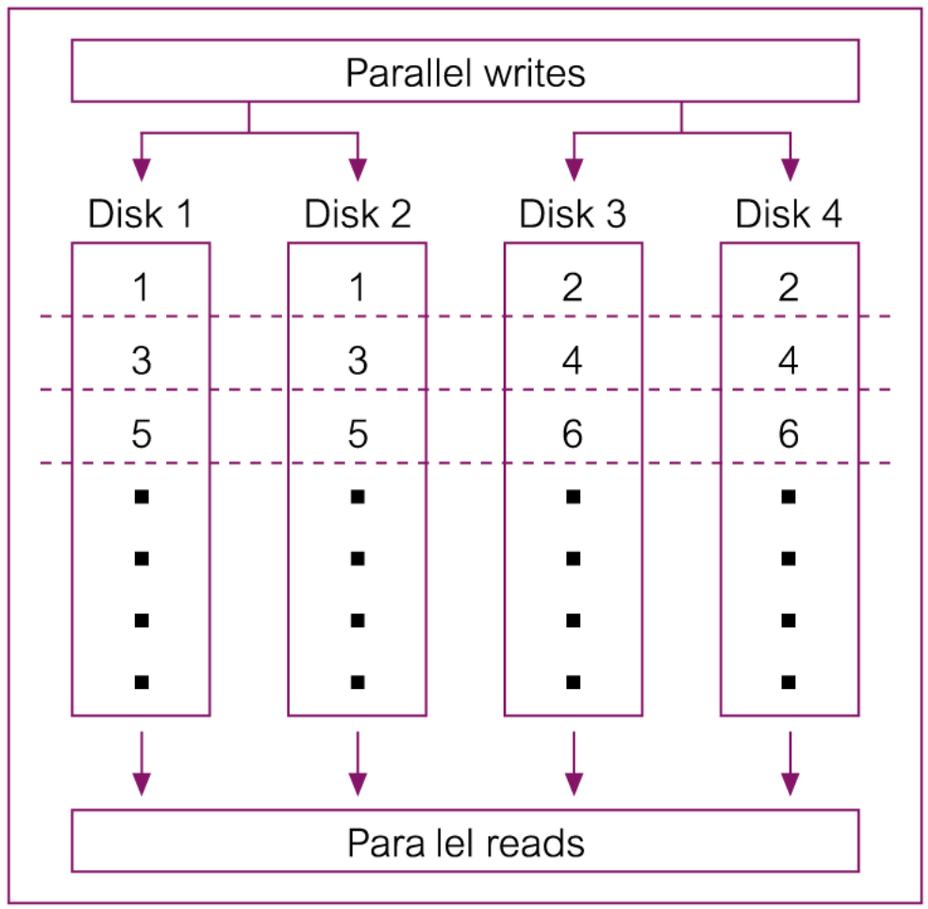
RAID Technology

- RAID Levels:
  - RAID 0 Non-redundant
  - RAID 1 Mirrored
  - RAID 2 Memory-Style Error-Correcting Codes
  - RAID 3 Bit-Interleaved Parity
  - RAID 4 Block-Interleaved Parity
  - RAID 5 Block-Interleaved Distributed Parity
  - RAID 0+1 Non-redundant and Mirrored

RAID 0 and RAID 1

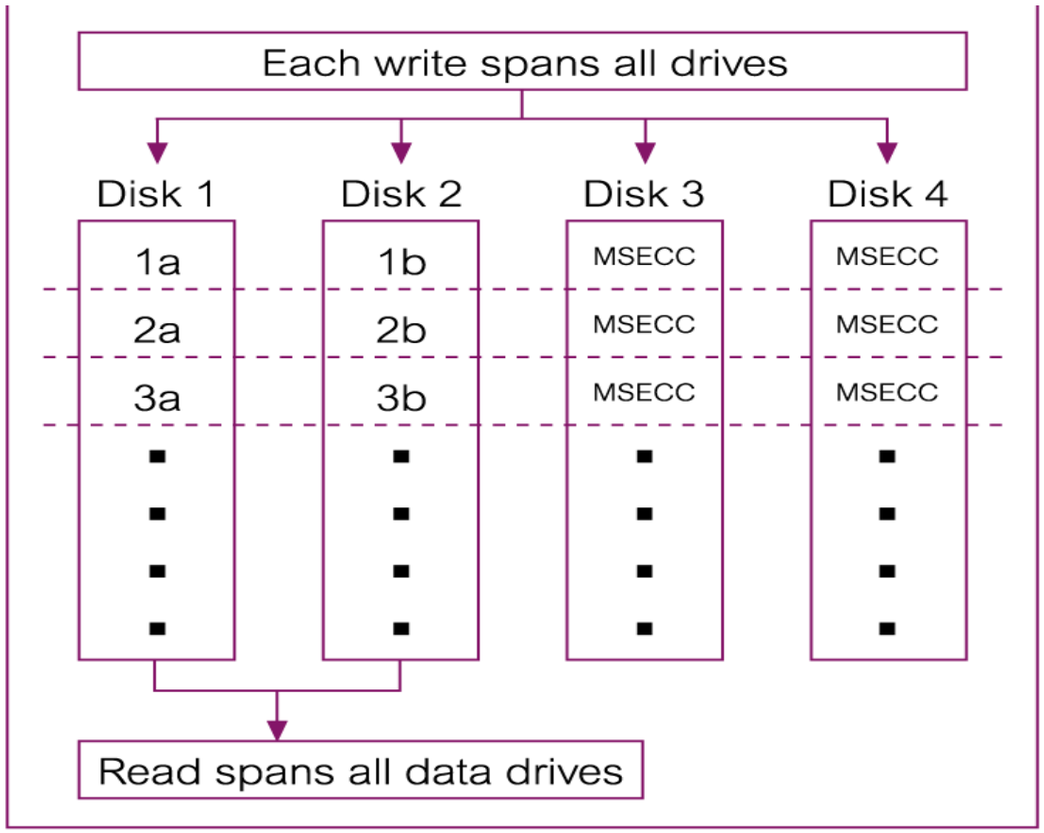


(a) RAID 0 – Nonredundant

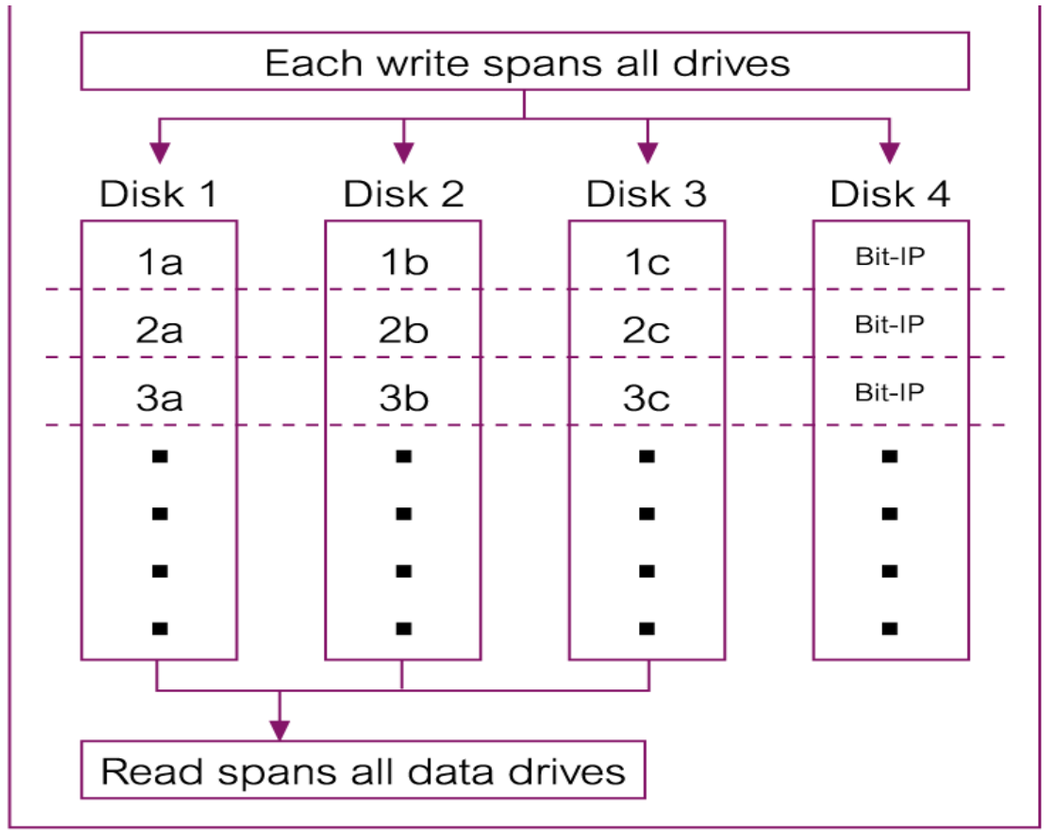


(b) RAID 1 – Mirrored

RAID 2 and RAID 3

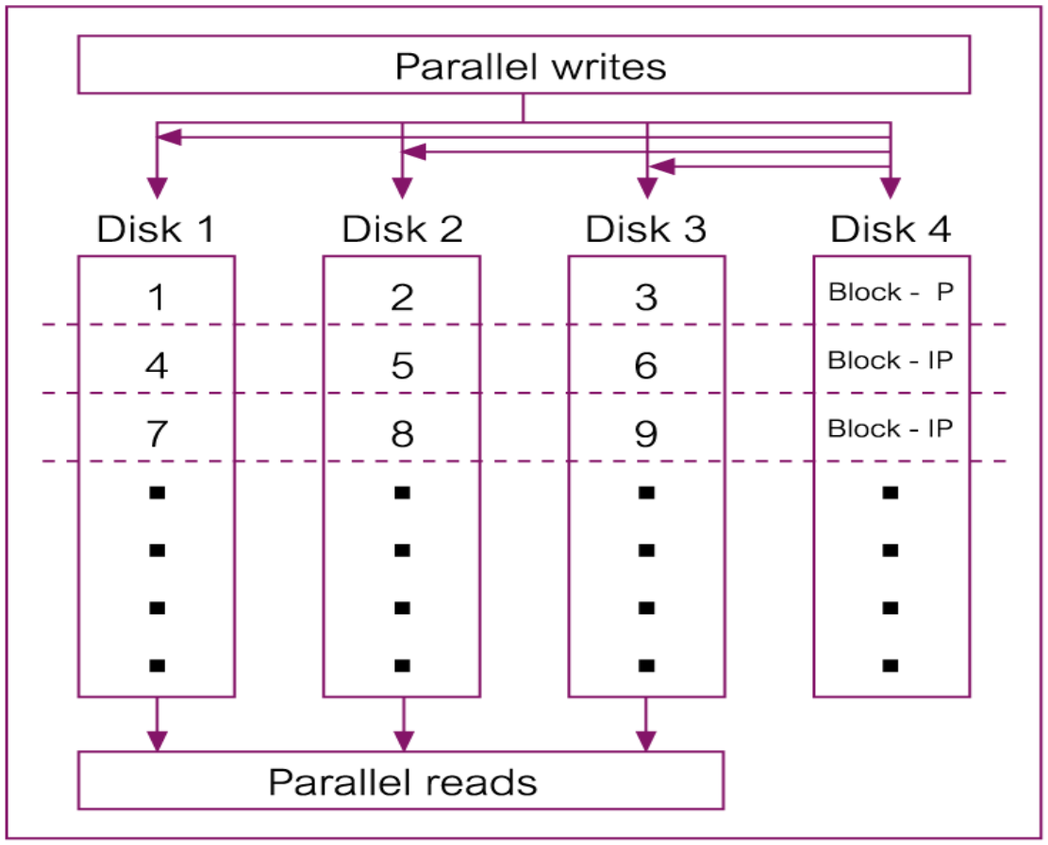


(c) RAID 2 – Memory-Style Error-Correct ng Codes (MSECC)

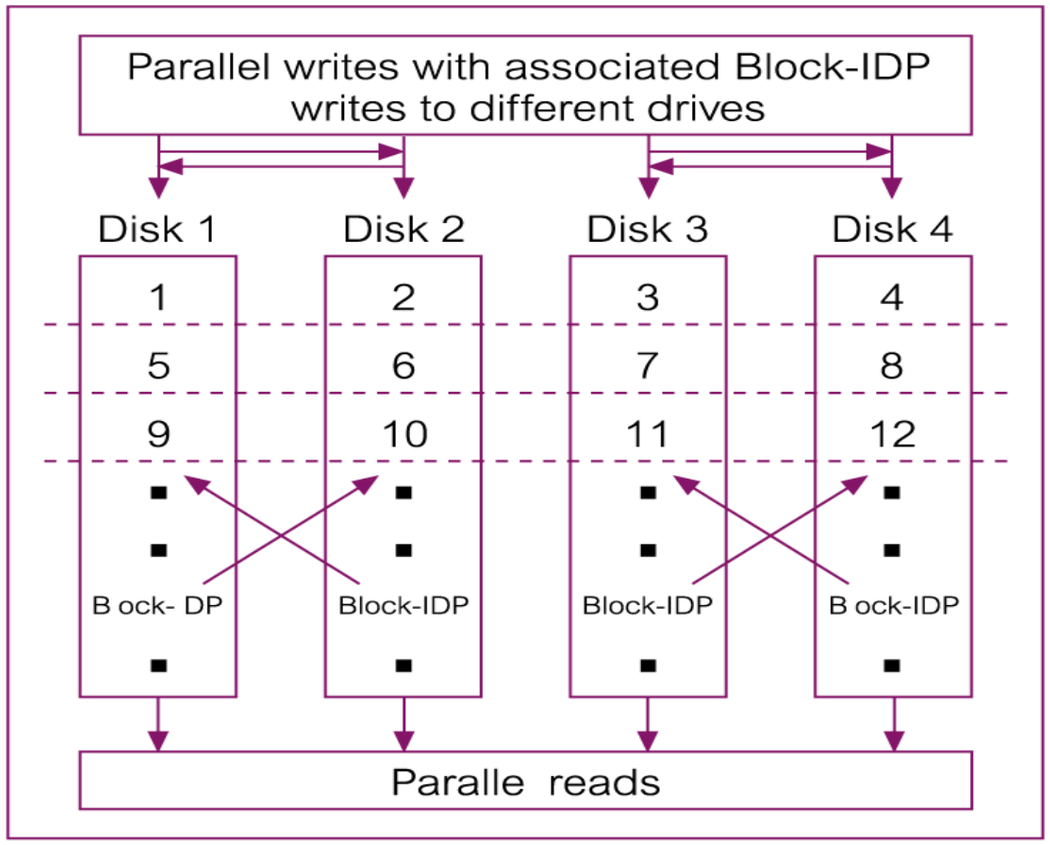


(d) RAID 3 – Bit Interleaved Parity (Bit-IP)

RAID 4 and RAID 5



(e) RAID 4 – Block-Interleaved Parity (Block-IP)



(f) RAID 5 – Block-Interleaved D str buted Parity (Block-IDP)