

Chapter 28: MySQL Backup & Recovery

28.1 MySQL Backup Concepts

- * Backup is the process of creating a copy of the data to protect it from **data loss**.
- * It helps to **recover** data in case of failure, corruption, or accidental deletion.
- * MySQL provides **different types** of backup methods.
- * A good backup strategy ensures **data safety** and business continuity.



Types of Backup:

Backup Type	Description
Logical Backup	Backs up the database objects (tables, data, etc.) in a human-readable format (SQL) .
Physical Backup	Backs up the actual data files at the storage level.
Full Backup	Backs up the entire database .
Incremental Backup	Backs up only the changes made since the last backup.
Differential Backup	Backs up the changes made since the last full backup .



Key Points :

- ✓ Backups **protect data** from loss or damage.
- ✓ Use both **logical** and **physical** backups for better security.
- ✓ Always **verify** backups by restoring in a test environment.
- ✓ Keep **multiple backup copies** and store them **securely**.



28.2 Logical Backup

- * Logical backup creates a backup in the form of **SQL statements**.
- * The backup file is **human-readable** and can be edited if needed.
- * It is **portable** across different systems and MySQL versions.
- * MySQL provides the **mysqldump** utility to perform logical backups.



Syntax :

```
mysqldump -u username -p [options] database_name > backup_file.sql;
```

Examples :

- `mysqldump -u root -p mydb > mydb_backup.sql;`
- `mysqldump -u root -p --all-databases > all_backup.sql;`
- `mysqldump -u root -p mydb table1 table2 > tables_backup.sql;`
- `mysqldump -u root -p --routines --events mydb > full_backup.sql;`

Options :

- databases db_name : Backup a database
- all-databases : Backup all databases
- tables tbl1 tbl2 : Backup specific tables
- routines : Include stored routines
- events : Include events

Key Points :

- ✓ Logical backup is done using **mysqldump**.
- ✓ Backup files are in **.sql** format and are easy to restore.
- ✓ It is **flexible**, **portable** and user-friendly.
- ✓ Best for **small to medium** databases.

Advantages of Logical Backup :

- Human-readable and editable
- Easy to restore
- Works across different platforms and versions
- Takes less space for smaller databases



Key Takeaway :

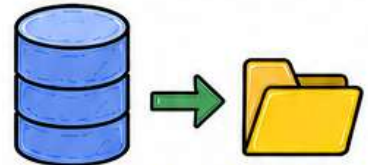


Logical backups using **mysqldump** are **simple**, **flexible** and **portable**. They are very useful for **managing** and **recovering** MySQL databases.



28.3 Physical Backup

- * Physical backup copies the **actual data files** at the storage level.
- * It is **faster** than logical backup and is used for large databases.
- * The backup is **not human-readable**.
- * It requires **more storage space**.
- * Best for **disaster recovery** and full backup.



PHYSICAL BACKUP

Methods of Physical Backup:

Method	Description
File System Copy	Copy the data directory (datadir) of MySQL while the server is stopped.
LVM Snapshot	Use Logical Volume Manager to create a snapshot of the MySQL data volume.
Storage Snapshot	Use storage system (SAN/NAS) snapshots to create a backup of data files.
MySQL Enterprise Backup	Percona XtraBackup or MySQL Enterprise Backup tools for hot backups .



When to Use ?

- Large databases
- High performance requirements
- Disaster recovery
- When quick restore of entire database is needed

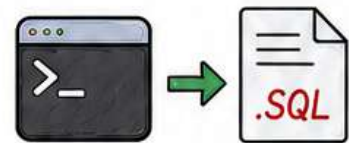
Key Points :

- ✓ Physical backup copies **actual data files**.
- ✓ It is **not** in SQL format.
- ✓ It is **faster** but **not portable**.
- ✓ Always **stop MySQL server** before taking file system copy.
- ✓ Best suited for **full backup** and **recovery**.



28.4 mysqldump

- * mysqldump is a MySQL utility used to create **logical backups**.
- * It exports the database structure and data as **SQL statements**.
- * The backup file can be used to **restore** the database.
- * It is part of MySQL and works on **all major platforms**.



mysqldump

Syntax :

```
mysqldump -u username -p [options] database_name > backup_file.sql;
```

Examples :

- `mysqldump -u root -p mydb > mydb_backup.sql;` -- Backup database mydb
- `mysqldump -u root -p --all-databases > all_backup.sql;` -- Backup all databases
- `mysqldump -u root -p mydb table1 table2 > tables_backup.sql;` -- Backup specific tables
- `mysqldump -u root -p --routines --events mydb > full_backup.sql;` -- Backup with routines and events
- `mysqldump -u root -p --no-data mydb > structure_only.sql;` -- Backup only structure

Key Points :

- ✓ mysqldump creates backups in **SQL format**.
- ✓ The backup file is **human-readable** and editable.
- ✓ It is **portable** and easy to restore.
- ✓ Include **--routines** and **--events** for complete backup.
- ✓ Use **--single-transaction** for InnoDB tables.



Common Options :

- **--all-databases** : Backup all databases
- **--no-data** : Backup structure only
- **--routines** : Include stored routines
- **--events** : Include events
- **--single-transaction** : Consistent backup for InnoDB tables

Key Takeaway :



mysqldump is the most widely used tool for logical backups.
It is **simple**, **flexible** and **safe** for most backup and migration tasks.



28.5 mysqlpump Concepts

- * mysqlpump is a **logical backup** utility introduced in MySQL 5.7.
- * It is an **improved version** of mysqldump.
- * It supports **parallel processing** to speed up the backup.
- * It provides **consistent** and **reliable** backups for large databases.
- * It can back up **databases, schemas, tables, routines, events, users** and **privileges**.



Syntax :

```
mysqlpump -u username -p [options] database_name > backup_file.sql;
```

Common Options :

Option	Description
--all-databases	Backup all databases.
--databases db1 db2	Backup specified databases.
--tables tbl1 tbl2	Backup specific tables.
--routines	Include stored routines.
--events	Include events.
--users	Include users and privileges.
--default-parallelism=N	Use N threads for parallel backup.
--compress-output	Compress the output file.
--exclude-databases=db1	Exclude specified databases.

Advantages of mysqlpump :

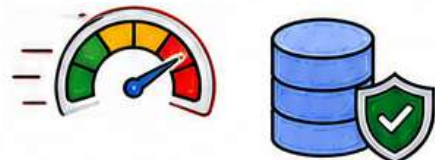
- Faster backup using parallel processing.
- Better performance for large databases.
- Supports more object types (users, roles, events, etc.).
- More consistent and reliable backups.
- Easy to use and flexible.

Examples :

- `mysqlpump -u root -p mydb > mydb_backup.sql;` -- Backup database mydb
- `mysqlpump -u root -p --all-databases > all_backup.sql;` -- Backup all databases
- `mysqlpump -u root -p --routines --events mydb > routine_event_backup.sql;` -- Backup routines and events
- `mysqlpump -u root -p --default-parallelism=4 mydb > mydb_parallel.sql;` -- Parallel backup with 4 threads

Key Points :

- ✓ mysqlpump is **faster** and more **efficient** than mysqldump.
- ✓ It supports **parallelism** and more object types.
- ✓ It is suitable for **large databases** and production environments.
- ✓ It provides **consistent** and **reliable** logical backups.



28.6 Database Restoration

- * Restoration is the process of **recovering data** from a backup.
- * It is used when data is **lost, corrupted** or **accidentally deleted**.
- * You can restore from both **logical backup** and **physical backup**.
- * Always make sure to **take a backup** before restoration.

Restoration Methods :

Backup Type	Restoration Method
Logical Backup	Use mysql command to import SQL file.
Physical Backup	Copy data files back to data directory.
Full Backup	Restore entire database from backup.
Incremental Backup	Apply base backup and then incremental changes.

Restoration Commands :

Task	Command
Restore Logical Backup	<code>mysql -u username -p database_name < backup_file.sql;</code>
Restore All Databases	<code>mysql -u root -p < backup_file.sql;</code>
Restore Specific Database	<code>mysql -u root -p db_name < backup_file.sql;</code>
Restore Physical Backup	Copy files to datadir and start MySQL server.



Important Tips :

- Always stop MySQL server for physical restore.
- Use the same MySQL version for restoration.
- Verify the backup before restoration.
- Test the restoration in a safe environment.

Logical Restore Example :

```
mysql -u root -p mydb < mydb_backup.sql;
-- Restore database mydb
mysql -u root -p < all_backup.sql;
-- Restore all databases
```

Key Takeaway :

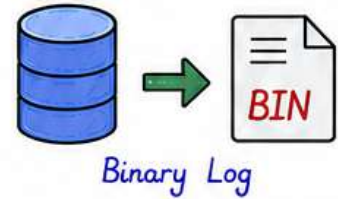


Regular backups and successful restorations are essential for **data safety** and **business continuity**. Always **test** your backups and **recovery** process.



28.7 Binary Logs

- * Binary log records **all changes** made to the database.
- * It is used for **replication, backup** and **recovery**.
- * The binary log contains events like **INSERT, UPDATE, DELETE, DDL**, etc.
- * Binary logs are essential for **Point-in-Time Recovery (PITR)**.
- * Binary log is enabled using the **log_bin** system variable.



Binary Log Events :

Event Type	Description
Statement Events	Records SQL statements executed by users.
Row Events	Records changes at row level (before and after image).
DDL Events	Records data definition statements (CREATE, DROP, ALTER).
Transaction Events	Records transaction begin, commit and rollback.
Other Events	Records events like user changes, table map, etc.

 **Uses of Binary Logs :**

- Data replication (Master-Slave).
- Incremental backup.
- Point-in-Time Recovery.
- Audit of database changes.
- Disaster recovery.

Key Points :

- ✓ Binary logs record **all changes** made to the database.
- ✓ Enabling binary log is important for **recovery** and **replication**.
- ✓ Keep binary logs for a **sufficient time** for recovery.
- ✓ Manage binary logs to save **disk space**.

Syntax to enable Binary Log :

In my.cnf / my.ini
`log_bin = mysql-bin`
`server_id = 1`
`binlog_format = ROW`
Then restart MySQL server.



Show Binary Logs :
`SHOW BINARY LOGS;`

Key Takeaway :

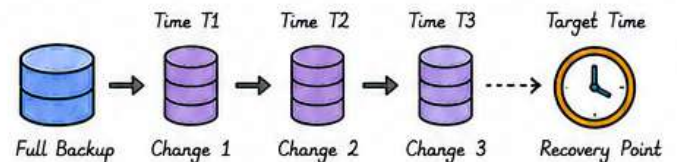


Binary logs are the backbone of MySQL **replication, backup** and **Point-in-Time Recovery**.
Always enable and manage binary logs properly for **data safety**.



28.8 Point-in-Time Recovery

- * Point-in-Time Recovery (PITR) allows you to restore the database to a **specific point in time**.
- * It uses **binary logs** to roll back or roll forward transactions.
- * PITR requires a **full backup** and **binary logs**.
- * Useful for recovering from **user errors, data corruption** or **accidental deletion**.



Pisarcia Process :

1. Take Full Backup
Take a full backup of the database.

2. Enable Binary Logs
Ensure binary logs are enabled and available.

3. Identify Target Time
Determine the exact time to which you want to recover.

4. Restore Backup
Restore the full backup on the server.

5. Apply Binary Logs
Apply binary logs up to the target time.

Method	Description
ROLLBACK (Stop at Time)	Apply binary logs until just before the target time.
ROLLFORWARD (Up to Time)	Apply binary logs until the target time is reached.
mysqlbinlog Utility	Used to read and apply binary logs.
STOP-DATETIME Option	Used with mysqlbinlog to specify stop time.

Example Command :

```
mysqlbinlog --start-datetime="2024-05-20 10:00:00" \  
--stop-datetime="2024-05-20 12:30:00" \  
--database=mydb mysql-bin.0001 | mysql -u root -p
```

Key Points :

- ✓ PITR requires **full backup + binary logs**.
- ✓ Always **keep binary logs** until you are sure backups are not needed.
- ✓ **Test recovery** process regularly.
- ✓ Ensure **server time** is accurate.

Best Practices :

- Automate backups and log retention.
- Store backups and logs in a secure location.
- Monitor disk space for binary logs.
- Regularly test Point-in-Time Recovery.



Key Takeaway :

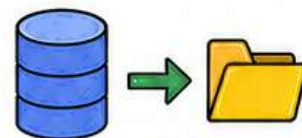


Point-in-Time Recovery helps you recover data to any exact moment using binary logs.
It is a powerful feature for **data protection** and **disaster recovery**.



28.9 Backup Strategies

- * A good backup strategy **reduces data loss and recovery time**.
- * It defines **what to backup**, **how often**, **where to store** and **how to restore**.
- * Common strategies include **Full**, **Incremental**, **Differential** and **Combined Backup**.
- * Follow the **3-2-1 rule** for best data protection.



Backup Strategy

Common Backup Strategies :

Strategy	Description	Advantages	Disadvantages	Best Use
Full Backup	Backup entire database at a point in time.	- Simple to restore. - Complete data.	- Takes more time. - Uses more storage.	Weekly / Monthly backup.
Incremental Backup	Backup only changes since the last backup (of any type).	- Fast backup. - Less storage.	- Restore is complex. - Needs all previous backups.	Daily backup after full backup.
Differential Backup	Backup changes since the last full backup.	- Easier restore than incremental. - Less storage than full.	- Backup size grows with time. - More storage than incremental.	Daily backup after full backup.
Combined Strategy	Use full backup + incremental backups.	Balanced in time and storage.	More management required.	Large databases in production.



3-2-1 Backup Rule :

- 3 copies of your data.
- 2 different types of storage media.
- 1 copy stored off-site (or in cloud).

Backup Locations :

- Local Disk
- External Hard Drive
- Network Share (NAS)
- Cloud Storage

Key Points :

- ✓ Choose a strategy based on **RTO** and **RPO** requirements.
- ✓ Test backups **regularly**.
- ✓ **Automate** backups to avoid human errors.
- ✓ Monitor backup **size**, **time** and **storage usage**.
- ✓ Verify backups by performing **test restores**.



RTO & RPO :

RTO (Recovery Time Objective) :
Maximum acceptable time to restore systems.

RPO (Recovery Point Objective) :
Maximum acceptable data loss measured in time.



Key Takeaway :

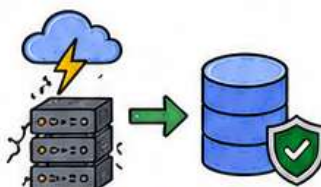


A well-planned backup strategy ensures **business continuity**, **minimizes data loss** and makes restoration **faster** and **easier**.



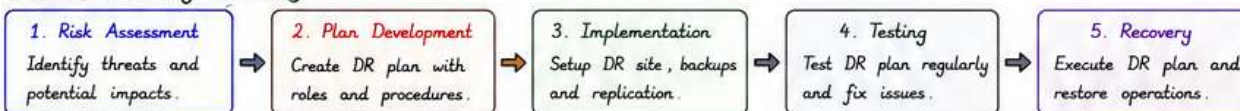
28.10 Disaster Recovery

- * Disaster Recovery (DR) is the process of restoring systems and data after a **major failure or disaster**.
- * It ensures **business continuity** and **minimizes downtime**.
- * DR plan includes **people**, **processes**, **technology** and **communication**.
- * It is **different from backup**, but uses backups as a key component.



Disaster Recovery

Disaster Recovery Process :



DR Strategies :

Strategy	Description	Advantages	Disadvantages	Best Use
Backup & Restore	Restore data from backups in case of disaster.	- Low cost. - Easy to implement.	- Longer recovery time. - Higher data loss.	Small businesses with low RPO.
Pilot Light	Minimal resources running at DR site.	- Lower cost. - Faster recovery than backup & restore.	- Manual setup needed. - Some data loss.	Medium businesses.
Warm Standby	Partial resources running at DR site.	- Faster recovery. - Less data loss.	- Higher cost. - Complex setup.	Large businesses.
Hot Standby	Full system running at DR site.	- Very fast recovery. - No data loss.	- Very high cost. - High maintenance.	Mission critical applications.

- ✓ Have a documented DR plan.
- ✓ Test it regularly.
- ✓ Store backups off-site or in cloud.
- ✓ Communicate clearly during disasters.



Key Takeaway :



A strong disaster recovery plan ensures **quick recovery**, **reduces downtime** and **protects data and business operations**.



Important Terms :

- **RTO (Recovery Time Objective) :**
Time within which the system must be restored.
- **RPO (Recovery Point Objective) :**
Maximum data loss acceptable.
- **DR Site :** Secondary site used for recovery.
- **Failover :** Switching operations to DR site.

DR Best Practices :

- ✓ Keep DR plan up to date.
- ✓ Test DR plan regularly.
- ✓ Document all procedures.
- ✓ Train teams and assign roles.
- ✓ Use automated monitoring.

