

MySQL Backup & Recovery Reference

Production guide to mysqldump, MySQL Shell, XtraBackup, binlog archiving, PITR and disaster scenarios

 RPO

How much data? · acceptable loss

 RTO

How fast? · time to restore

 RULE

3-2-1-1-0 · 3 copies, 2 media, 1 off-site

 +

imm

01

Fundamentals & Strategy



CONCEPT	DESCRIPTION
Logical	SQL / object-level dump. Version & platform independent, easy selective restore; slow restore on large data (indexes are rebuilt).
Physical	Block copy of the data files. Fast restore , fits TB scale; requires same major version + platform.
Snapshot	LVM / ZFS / EBS / SAN snapshot. Crash-consistent; safe if WAL/redo is on the same volume or backup mode is used.
Full / Incr / Diff	Everything / changes since last backup / since last full. The longer the incremental chain, the higher restore time and risk .
Hot / Warm / Cold	Reads+writes allowed / reads only / server stopped. Production target: hot .
Continuous Archive	Full backup + continuous binlog stream → PITR : return to any moment. Brings RPO down to seconds.



CHOOSING A METHOD

SCENARIO	RECOMMENDATION
< 100 GB, version migration, single schema	util.dumpSchemas / mydumper
TB scale, short RTO	XtraBackup / MEB + binlog
RPO in seconds	Semi-sync replica + binlog stream
"I deleted it by mistake" protection	PITR + SOURCE_DELAY



TOOL MAP

Logical	mysqldump · MySQL Shell util.* · mydumper
Physical	XtraBackup · MEB (mysqlbackup) · CLONE
Binlog	mysqlbinlog · SHOW BINARY LOGS
InnoDB Only	Hot physical backup and consistent dumps are guaranteed only for InnoDB



Replication is not a backup: DROP TABLE or a bad UPDATE reaches every replica within milliseconds. HA ≠ DR.

02

Logical Backup

TOOL	PARALLEL	NOTE
mysqldump	X	Available everywhere; single thread, small/medium
mysqlpump	✓	deprecated in 8.0.34, removed in 8.4 — avoid
Shell util.*	✓	dumpInstance / dumpSchemas / dumpTables + loadDump ; zstd, chunking, resume
mydumper	✓	Open source, parallel restore with myloader



MYSQLDUMP — PRODUCTION FLAGS

```
$ mysqldump --single-transaction --source-data=2 \
--routines --events --triggers --hex-blob \
--set-gtid-purged=ON --all-databases \
| zstd -T0 > /bk/full_$(date +%F).sql.zst
# --source-data=2 → binlog position written as a comment
# (before 8.0.26: --master-data=2)
```



MYSQL SHELL DUMP & LOAD (JS)

```
util.dumpInstance('/bk/inst_2026-10-13',
  {threads: 8, compression: 'zstd', consistent: true})
util.dumpSchemas(['shop', '/bk/shop', {threads: 8})
// target needs local_infile=ON
util.loadDump('/bk/inst_2026-10-13',
  {threads: 8, deferTableIndexes: 'all'})
```



RESTORE

```
$ zstd -dc full.sql.zst | mysql -u root -p
$ myloader -d /bk/mydumper -t 8 --overwrite-tables
```



Caution: `--single-transaction` is consistent only for **InnoDB**; MyISAM tables stay inconsistent. Do not run DDL (`ALTER/TRUNCATE`) during the dump — it breaks consistency.



Speed: Shell [util.loadDump](#) and [myloader](#) are much faster than a plain SQL restore: they load in parallel and build indexes last.

04

Binlog & Point-in-Time Recovery

CRITICAL

PARAMETER	RECOMMENDED
log_bin	ON (8.0+ default)
binlog_format	ROW
sync_binlog	1 — to disk on every commit
innodb_flush_log_at_trx_commit	1 — full durability
binlog_expire_logs_seconds	≥ age of the oldest backup
gtid_mode	ON + enforce_gtid_consistency

LIVE BINLOG BACKUP (ON A SEPARATE SERVER)

```
$ mysqlbinlog --read-from-remote-server --raw \
--stop-never --host=db1 --user=bkp -p \
--result-file=/binlogs/ binlog.000001
```

STEP BY STEP PITR

1

Restore the full backup (dump / XtraBackup / CLONE).

2

Find the start position: dump header or `xtrabackup_binlog_info`.

3

Find the position of the bad event (below).

4

Replay binlogs **in a single command** up to just before the event.

FIND THE EVENT & REPLAY

```
$ mysqlbinlog -v --base64-output=DECODE-ROWS \
--start-datetime="2026-10-13 09:50:00" binlog.000043 \
| grep -n -B5 -i "DROP TABLE" # → end_log_pos
$ mysqlbinlog --start-position=157 --stop-position=48211 \
binlog.000042 binlog.000043 | mysql -u root -p
# With GTID: skip the bad transaction, replay the rest
$ mysqlbinlog --exclude-gtids='3E11FA47-...:1234' \
binlog.000042 binlog.000043 | mysql -u root -p
```

⚠

Trap: Do not replay binlog files **one by one** with `mysqlbinlog` | `mysql` — temporary tables and transactions spanning files break. Pass them all in one call.

05

Verification, Testing & Monitoring

CHECK	TOOL / METHOD
Backup Integrity	xtrabackup --prepare → "completed OK!", MEB validate
Data Integrity	CHECK TABLE, mysqlcheck, CHECKSUM TABLE
Replica Consistency	pt-table-checksum → if drift found pt-table-sync
Binlog Health	binlog stream process, SHOW BINARY LOGS, gtid_purged
Restore Test	Automated, at least weekly; restore on a separate server → validate with queries → measure RTO and record it.

BINLOG HEALTH CHECK

```
SHOW BINARY LOGS;
SHOW BINARY LOG STATUS; -- 8.4 (8.0: SHOW MASTER STATUS)
SELECT @@gtid_executed, @@gtid_purged,
@@binlog_expire_logs_seconds;
```

INTEGRITY CHECKS

```
$ mysqlcheck --all-databases --check -u root -p
$ mysqlbackup --backup-image=/bk/full.mbi validate
$ pt-table-checksum --replicate=percona.checksums h=db1
CHECKSUM TABLE shop.orders;
```

SET UP ALERTS

Age of last successful backup > 26 hours

CRITICAL

Binlog stream process down / lag > RPO

CRITICAL

Binlog purged before the next backup

CRITICAL

Sudden ±30% change in backup size

IMPORTANT

Backup disk usage > 80%

IMPORTANT

ⓘ

MySQL 8.4: SHOW MASTER STATUS was removed → use SHOW BINARY LOG STATUS (8.2+). Update monitoring scripts before upgrading.