Backup & Recovery Techniques

6.1 Backup Requirements and Strategy

RPO and RTO

RPO (Recovery Point Objective): Maximum acceptable data loss

- Example: RPO 1 hour = Can lose up to 1 hour of data
- Determined by backup frequency and strategy

RTO (Recovery Time Objective): Maximum acceptable recovery time

- Example: RTO 30 minutes = Must restore within 30 minutes
- Determined by backup size and complexity

Backup Types

Full Backup:

- Backs up entire database
- Time-consuming, large file size
- Easiest to restore
- Recommended: Daily or weekly

Incremental Backup:

- Only changes since last backup
- Fast, small file size
- Requires full backup + all increments to restore
- Recommended: Daily

Differential Backup:

- Changes since last full backup
- Medium speed and size
- Requires full backup + one differential
- Recommended: Between full backups

Backup Methods

Hot Backup:

- Database remains online during backup
- Minimal performance impact
- Methods: Binary log + InnoDB backup, Percona XtraBackup
- Recommended for production

Cold Backup:

- Database offline during backup
- No performance impact to other operations
- Complete data consistency guaranteed
- Methods: File system backup, mysqldump

Logical Backup:

- SQL statements that recreate data
- Human-readable format
- Slower than physical backup
- Portable across servers
- Method: mysqldump

Physical Backup:

- Raw database file copy
- Fast, compact
- Not portable between architectures
- Methods: XtraBackup, file system copy

6.2 Backup with mysqldump

Basic Backup Operations

```
# Full database backup

mysqldump -u root -p --all-databases > full_backup.sql
```

```
# Specific database
mysqldump -u root -p database_name > db_backup.sql

# Specific table
mysqldump -u root -p database_name table_name > table_backup.sql

# Multiple tables
mysqldump -u root -p database_name table1 table2 > tables_backup.sql

# Multiple databases
mysqldump -u root -p --databases db1 db2 db3 > multi_db_backup.sql
```

Advanced Backup Options

```
# Consistent snapshot for InnoDB (recommended)
mysqldump -u root -p \
  --single-transaction \    # Consistent snapshot without locking
  --quick \                 # Uses WHERE limit for faster backup
  --lock-tables=false \     # No table locks
  --all-databases > backup.sql

# Complete backup with all objects
mysqldump -u root -p \
  --triggers \              # Include triggers
  --routines \              # Include stored procedures/functions
  --events \                # Include scheduled events
  --comments \              # Include helpful comments
  database_name > complete_backup.sql
```

```
# Backup with binary log position
mysqldump -u root -p \
  --master-data=2 \        # Include CHANGE MASTER statement
  --all-databases > backup_with_position.sql
```

Compression and Encryption

```
# Gzip compression
mysqldump -u root -p database_name | gzip > backup.sql.gz

# Bzip2 compression
mysqldump -u root -p database_name | bzip2 > backup.sql.bz2

# Encryption with openssl
mysqldump -u root -p database_name | \
  openssl enc -aes-256-cbc -S 8 -out backup.sql.enc

# Encryption with GPG
mysqldump -u root -p database_name | \
  gpg --encrypt --recipient user@example.com > backup.sql.gpg
```

Restoring from mysqldump Backup

```
# Restore full database
mysql -u root -p < backup.sql

# Restore specific database
mysql -u root -p database_name < db_backup.sql
```

```
# Restore compressed backup
gunzip < backup.sql.gz | mysql -u root -p

# Restore with error handling
mysql -u root -p --force < backup.sql  # Continue on error

# Restore specific part of backup
head -100 backup.sql | mysql -u root -p  # First 100 lines
grep "INSERT INTO users" backup.sql | mysql -u root -p database_name
```

6.3 Percona XtraBackup

Installation

```
# Ubuntu/Debian
sudo apt install percona-xtrabackup-24  # MySQL 5.7
sudo apt install percona-xtrabackup-80  # MySQL 8.0

# CentOS/RedHat
sudo yum install percona-xtrabackup-24
sudo yum install percona-xtrabackup-80

# Verify
xtrabackup --version
```

Full Backup

```
# Create backup directory
mkdir -p /backup/full
chmod 700 /backup/full
```

```
# Perform full backup
xtrabackup --backup \
    --target-dir=/backup/full \
    --user=root \
    --password=YourPassword

# Prepare backup (make consistent)
xtrabackup --prepare --target-dir=/backup/full

# Verify backup
ls -la /backup/full/
```

Incremental Backup

```
# After full backup, create incremental
mkdir -p /backup/incremental_1
xtrabackup --backup \
    --target-dir=/backup/incremental_1 \
    --incremental-basedir=/backup/full \
    --user=root \
    --password=YourPassword

# Make database changes...
# Create second incremental
mkdir -p /backup/incremental_2
xtrabackup --backup \
   --target-dir=/backup/incremental_2 \
   --incremental-basedir=/backup/incremental_1 \
   --user=root \
   --password=YourPassword

# Prepare incremental backups
xtrabackup --prepare --apply-log-only --target-dir=/backup/full
xtrabackup --prepare --apply-log-only \
   --target-dir=/backup/full \
   --incremental-dir=/backup/incremental_1
xtrabackup --prepare --target-dir=/backup/full \
   --incremental-dir=/backup/incremental_2
```

Compressed Backup

```
# Backup with compression
xtrabackup --backup \
   --target-dir=/backup/compressed \
   --compress \             # Enable compression
   --compress-threads=4 \   # Parallel compression
   --user=root \
   --password=YourPassword

# Decompress before prepare
xtrabackup --decompress --target-dir=/backup/compressed
```

```
# Prepare compressed backup
xtrabackup --prepare --target-dir=/backup/compressed
```

Restoring from XtraBackup

```
# Stop MySQL
sudo systemctl stop mysql

# Backup current datadir
sudo mv /var/lib/mysql /var/lib/mysql.backup

# Copy backup to datadir
xtrabackup --copy-back --target-dir=/backup/full

# Fix permissions
sudo chown -R mysql:mysql /var/lib/mysql
sudo chmod 750 /var/lib/mysql

# Start MySQL
sudo systemctl start mysql

# Verify restoration
mysql -u root -p -e "SELECT VERSION();"
```

Remote Backup

```
# Backup to remote server
xtrabackup --backup \
   --stream=xbstream \       # Stream format for piping
   --target-dir=./ \
   --user=root \
   --password=YourPassword | \
   ssh user@remote "xbstream -x -C /backup/"

# Prepare remote backup
ssh user@remote "xtrabackup --prepare --target-dir=/backup"

# Restore remote backup
ssh user@remote "xtrabackup --copy-back --target-dir=/backup"
```

6.4 Point-in-Time Recovery (PITR)

Prerequisites

Binary logs must be enabled:

```
[mysqld]
log_bin = /var/log/mysql/mysql-bin
server_id = 1
```

PITR Procedure

Step 1: Take baseline backup

```
mysqldump -u root -p --single-transaction \
   --all-databases > backup.sql
```

```
# Note backup timestamp and binary log position
mysql -u root -p -e "SHOW MASTER STATUS;" > backup_position.txt
```

Step 2: Make database changes

```
# Record current time
date  # e.g., 2026-01-14 16:30:45

# Make database changes
mysql -u root -p database_name -e "INSERT INTO data VALUES (...);"
mysql -u root -p database_name -e "UPDATE table1 SET col1 = 'value';"
mysql -u root -p database_name -e "DELETE FROM table2 WHERE id > 100;"

# Accidentally delete data!
mysql -u root -p database_name -e "DELETE FROM important_table WHERE id > 0;"
```

Step 3: Identify recovery point

```
# Extract binary log contents
mysqlbinlog /var/log/mysql/mysql-bin.000001 > binlog.sql

# Find the DELETE statement
grep -n "DELETE FROM important_table" binlog.sql

# Alternative: by timestamp
mysqlbinlog \
   --start-datetime="2026-01-14 16:30:00" \
   --stop-datetime="2026-01-14 16:35:00" \
   /var/log/mysql/mysql-bin.000001 > recovered_changes.sql
```

Step 4: Recover to point-in-time

```
# Method 1: Using position
# Find position just before DELETE
mysqlbinlog \
  --stop-position=123456 \
  /var/log/mysql/mysql-bin.000001 | \
  mysql -u root -p

# Method 2: Using timestamp
mysqlbinlog \
  --stop-datetime="2026-01-14 16:34:59" \
  /var/log/mysql/mysql-bin.000001 | \
  mysql -u root -p

# Verify recovery
mysql -u root -p -e "SELECT COUNT(*) FROM important_table;"
```

PITR with Multiple Binary Log Files

```
# Recovery across multiple binary log files
mysqlbinlog \
  --stop-datetime="2026-01-14 16:34:59" \
  /var/log/mysql/mysql-bin.000001 \
  /var/log/mysql/mysql-bin.000002 \
  /var/log/mysql/mysql-bin.000003 | \
  mysql -u root -p
```

6.5 Backup Strategy for Production

Daily Backup Schedule

```
#!/bin/bash
# Daily backup script

BACKUP_DIR="/backups/mysql"
RETENTION_DAYS=7
DB_USER="root"
DB_PASS="password"
DATE=$(date +%Y%m%d_%H%M%S)

# Create daily backup
mkdir -p "$BACKUP_DIR/daily/$DATE"
mysqldump -u $DB_USER -p$DB_PASS \
   --single-transaction \
   --all-databases | \
   gzip > "$BACKUP_DIR/daily/$DATE/backup.sql.gz"

# Create binary log backup
cp /var/log/mysql/mysql-bin.* "$BACKUP_DIR/daily/$DATE/"

# Clean old backups
find "$BACKUP_DIR/daily" -type d -mtime +$RETENTION_DAYS -exec rm -rf {} \;

# Alert if backup failed
if [ $? -ne 0 ]; then
   echo "Backup failed!" | mail -s "MySQL Backup Alert" admin@example.com
fi
```

Full + Incremental Strategy

```
#!/bin/bash
# Weekly full backup, daily incremental

BACKUP_DIR="/backups/mysql"
DATE=$(date +%Y%m%d_%H%M%S)
DAY_OF_WEEK=$(date +%u)  # 1=Monday, 7=Sunday

if [ $DAY_OF_WEEK -eq 1 ]; then
  # Monday: Full backup
  xtrabackup --backup --target-dir=$BACKUP_DIR/full_$DATE
  echo $BACKUP_DIR/full_$DATE > /tmp/last_full_backup
else
  # Other days: Incremental backup
  LAST_FULL=$(cat /tmp/last_full_backup)
  xtrabackup --backup \
    --target-dir=$BACKUP_DIR/incr_$DATE \
    --incremental-basedir=$LAST_FULL
fi
```

6.6 Summary: Key Takeaways

1. **Backup Types**: Full, incremental, differential, hot, cold
2. **mysqldump**: Logical backup, portable, good for smaller databases
3. **Percona XtraBackup**: Physical backup, faster, better for large databases
4. **PITR**: Recover to specific point using binary logs
5. **Backup Strategy**:
    - RPO/RTO requirements determine strategy
    - Test recovery procedures regularly

- Retain backups for compliance

6. **Best Practices**:
    - Regular automated backups
    - Test restoration procedures
    - Store backups off-site
    - Document recovery procedures