
ORACLE®

MySQLバックアップ入門

Yoshiaki Yamasaki / 山崎 由章
MySQL Senior Sales Consultant, Asia Pacific and Japan

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Program Agenda

1. バックアップ手法の概要
2. バックアップ対象のファイル
3. バックアップ/リストア例
4. MySQL Enterprise Backup

Program Agenda

1. バックアップ手法の概要
2. バックアップ対象のファイル
3. バックアップ/リストア例
4. MySQL Enterprise Backup

バックアップ手法の違い

- ホットバックアップ or コールドバックアップ
- 物理バックアップ or 論理バックアップ
- フルバックアップ or 差分/増分バックアップ

- ※スタンバイサイトを利用する
  - ホットスワップとしての利用
  - バックアップ取得先としての利用

ホットバックアップ or コールドバックアップ

- ホットバックアップ(オンラインバックアップ)
  - データベースを停止せず、稼働したままでバックアップデータを取得する

- コールドバックアップ(オフラインバックアップ)
  - データベース停止中にバックアップデータを取得する

ホットバックアップ or コールドバックアップ

- ホットバックアップ(オンラインバックアップ)の例
  - トランザクションの仕組みを利用してバックアップを取得
    - mysqldumpでInnoDBテーブルをバックアップする
  - ロックを利用してバックアップを取得
    - mysqldumpでMyISAMテーブルをバックアップする
  - OSやハードウェアのスナップショットを利用する
    - LVMでスナップショットを取得してバックアップする(※)
  - 独自の方法でバックアップを取得する
    - MySQL Enterprise Backup(旧名称：InnoDB Hot Backup)でバックアップする(※)

※参考：漢(オトコ)のコンピュータ道 より
MySQLバックアップ頂上決戦！！ LVMスナップショット vs InnoDB Hot Backup
http://nippondanji.blogspot.jp/2009/12/mysql-lvm-vs-innodb-hot-backup.html

ホットバックアップ or コールドバックアップ

- コールドバックアップ(オフラインバックアップ)の例
  - MySQLサーバをシャットダウンして、データディレクトリ以下のディレクトリとファイルを全てOSのコマンドでコピー

物理バックアップ or 論理バックアップ

- 物理バックアップ
  - 物理的なファイルのバックアップ
  - OSファイルのコピー、MySQL Enterprise Backup等で取得可能
  - 利点：最小限のサイズで取得できる
    バックアップ/リストアの速度が速い
  - 欠点：バックアップ/リストアの単位はツールしだい
    異機種間、バージョン間で互換性が取れない場合がある(※)

※主な注意事項
- テーブル名に日本語文字を使用している場合
- 浮動小数点が混じっている場合
- テーブル名に大文字/小文字が混在している場合

物理バックアップ or 論理バックアップ

- 論理バックアップ
  - データベースからデータを抜き出してバックアップする
  - MySQLの場合は、SQLベースのバックアップを取得可能
  - mysqldumpで取得可能
  - 利点：バックアップファイルを編集できる
    移植性が高い（他バージョン、他のRDBMS）
  - 欠点：物理バックアップに比べてサイズが大きくなる
    バックアップ、リストアに時間がかかる(バイナリ⇔テキストの変換が入るため)

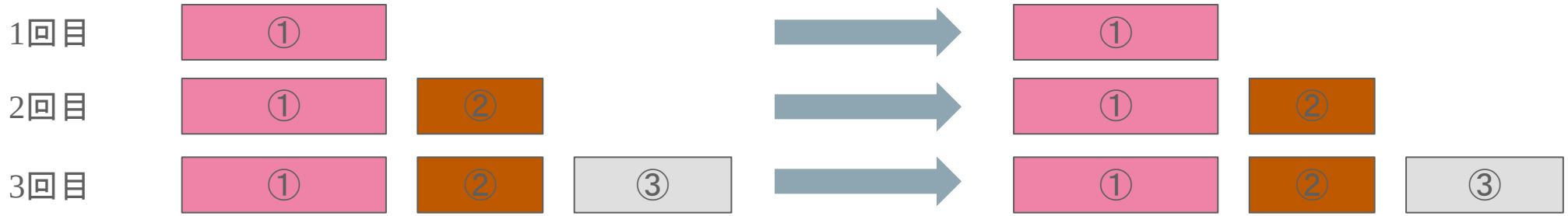
フルバックアップ or 差分/増分バックアップ

- フルバックアップ(全体バックアップ)
  - データベース全体をバックアップする

- 差分/増分バックアップ(部分バックアップ)
  - 直近のバックアップ以降に更新されたデータのみを
    バックアップする
  - 差分バックアップ
    - 直近のフルバックアップ以降に更新されたデータをバックアップ
  - 増分バックアップ
    - 直近のバックアップ(種別はフルとは限らない)以降に更新されたデータをバックアップする

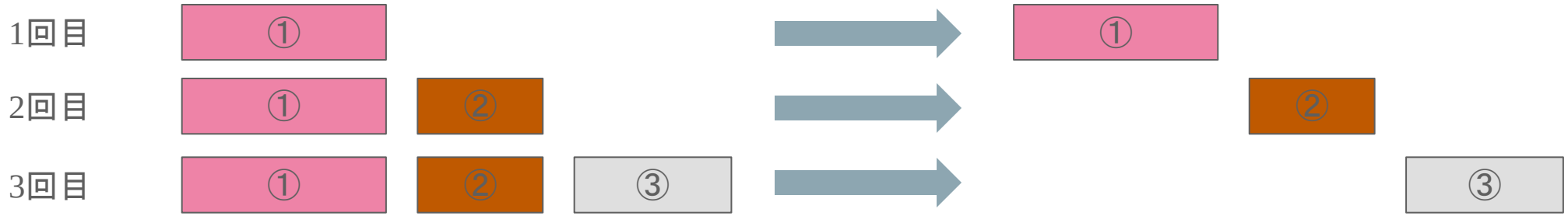
フルバックアップ or 差分/増分バックアップ

- フルバックアップ(全体バックアップ)
  - 利点
    - リストア処理が単純
      (バックアップデータが1カ所にまとまっているため)
  - 欠点
    - バックアップにかかる時間が長くなる (データベース全体をバックアップするため)
    - バックアップデータのサイズが大きくなる
      (データベースに対する更新量が少なくても、データベース全体をバックアップする)



フルバックアップ or 差分/増分バックアップ

- 差分/増分バックアップ(部分バックアップ)
  - 利点
    - バックアップ時間が短くなる (更新したデータだけを対象にするため)
    - バックアップデータのサイズが小さくなる (更新したデータだけを対象にするため)
  - 欠点
    - リストア処理の手順が複雑になる (フルバックアップと部分バックアップを使ってリストア)



フルバックアップ or 差分/増分バックアップ

- MySQLでの差分/増分バックアップ手法
  - バイナリログを増分バックアップとして利用する
  - MySQL Enterprise Backup(有償ツール)で InnoDBデータの増分/差分バックアップを取得する

ポイントインタイムリカバリとバイナリログ

- ポイントインタイムリカバリ
  - 特定の日時の状態にデータを復旧すること (例：障害発生直前の状態まで復旧する)
  - MySQLでは、バックアップファイルとバイナリログファイルを利用して、ポイントインタイムリカバリが可能
    (バックアップファイルに対して、バイナリログファイルを使って
    ロールフォワードリカバリする)
- バイナリログ
  - 発行されたクエリのうち、更新系のSQL文のみを記録しているログファイル
  - “--log-bin” オプションを設定することで出力できる
  - コミット時にバイナリログに同期書き込みするためには、”sync_binlog=1” を設定する

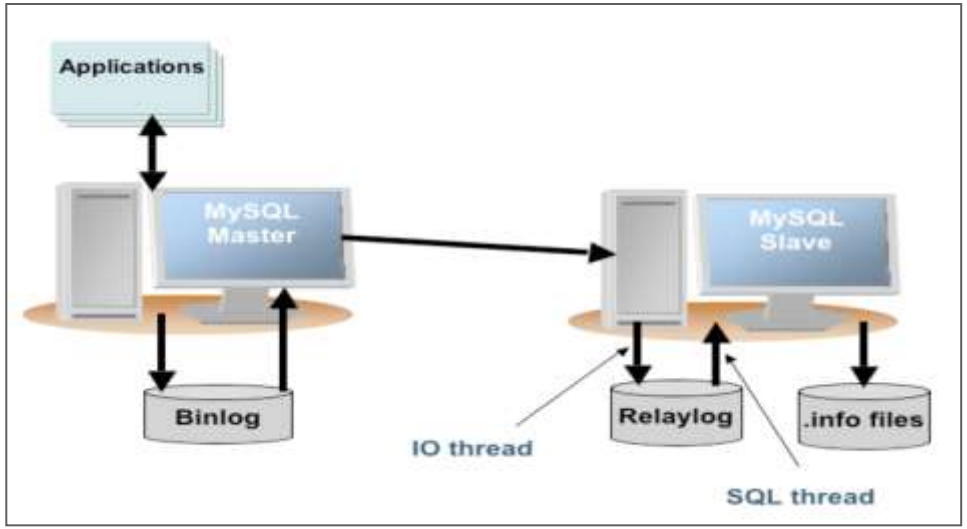
スタンバイサイトを利用する

- レプリケーション機能を利用して、スタンバイサイトを構築しておき、障害発生時はフェイルオーバーする
  - 利点
    - 復旧時間が短い
  - 欠点
    - 復旧ポイントは障害発生直前の状態のみ(※)
    - 人的ミスに対する対策にはならない(※)

    ※MySQL5.6では、遅延レプリケーション機能で、
    意図的にスタンバイサイトを遅延させることも可能
- スタンバイサイトをバックアップ取得先として利用することも可能
（本番環境に影響を与えずにバックアップ取得可能）

レプリケーション

- MySQLの標準機能
  - シンプルな設定で利用可能
  - 多数のWebサイトで実績あり
- 同期方式：非同期 or 準同期
- 特長
  - 参照性能を向上させる構成
  - バックアップ用途でも利用可能
  - バイナリログを利用して、
    更新内容をスレーブに伝搬



Program Agenda

1. バックアップ手法の概要
2. バックアップ対象のファイル
3. バックアップ/リストア例
4. MySQL Enterprise Backup

バックアップ対象

- データディレクトリ または データ全体
  - フルバックアップ
  - 論理 または 物理バックアップ
- ログファイル(バイナリログ)
  - 増分バックアップ
  - ポイントインタイムリカバリに利用
- 設定ファイル
  - `my.cnf`

データディレクトリ

- データベース内容、ログ、およびステータスファイルの格納先
- デフォルトのディレクトリはインストール形式による
  - **/usr/local/mysql/data/** (tar形式)
  - **/var/lib/mysql** (RPMパッケージ)
- サーバ起動オプションで設定可能
  - **datadir=/path/to/datadir/**
- サーバが利用しているディレクトリは下記コマンドで確認可能
  - **mysql> SHOW VARIABLES like 'datadir';**

※InnoDB関連ファイルのデータパスをデータディレクトリ以外に設定している場合は、バックアップ時にそれらも取得する

バイナリログ

- 発行されたクエリのうち、更新系のSQL文のみを記録しているログファイル
  - クエリ実行日時などのメタデータも記録
  - トランザクションのコミット時に同期的に記録 (**sync_binlog=1**)
- バイナリ形式で記録
  - **mysqlbinlog**コマンドにてテキスト化が可能
- 起動オプションを指定して、出力する
  - **--log-bin[=file_name]**
  - 通常の運用時には利用することを推奨
  - データディレクトリとは別のディスクに出力することを推奨
- ログファイル名の拡張子に通し番号を記録
  例）**file_name-bin.001, file_name-bin.002**, etc.
  - 現在利用中のログ番号はインデックスファイルに記録 (**file_name.index**)

バイナリログの管理

- **SHOW MASTER STATUS** コマンドで現在使用中のバイナリログファイル名とポジションを確認
- **SHOW MASTER LOGS** コマンドで全てのバイナリログファイル名を列挙
- **FLUSH [BINARY] LOGS** コマンドまたはMySQLサーバの再起動でログファイルのローテーション
- **PURGE MASTER** コマンドで特定の時点までのバイナリログを削除
- **RESET MASTER** コマンドで全てのバイナリログを削除

バイナリログの管理

```
mysql> SHOW MASTER STATUS;
+--------------+----------+--------------+------------------+
| File         | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+--------------+----------+--------------+------------------+
| MySQL.000007 |      107 |              |                  |
+--------------+----------+--------------+------------------+
1 row in set (0.00 sec)

mysql>
mysql> SHOW MASTER LOGS;
+--------------+-----------+
| Log_name     | File_size |
+--------------+-----------+
| MySQL.000001 |      1110 |
| MySQL.000002 |      2797 |
...
| MySQL.000007 |       107 |
+--------------+-----------+
7 rows in set (0.00 sec)
```

バイナリログの管理

```
mysql> FLUSH BINARY LOGS;
Query OK, 0 rows affected (0.42 sec)

mysql> SHOW MASTER STATUS;
+---------------+----------+--------------+------------------+
| File          | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+---------------+----------+--------------+------------------+
| MySQL.000008 |      107 |              |                  |
+---------------+----------+--------------+------------------+
1 row in set (0.00 sec)

mysql> SHOW MASTER LOGS;
+---------------+-----------+
| Log_name      | File_size |
+---------------+-----------+
| MySQL.000001 |      1110 |
...
| MySQL.000007 |       146 |
| MySQL.000008 |       107 |
+---------------+-----------+
8 rows in set (0.00 sec)
```

バイナリログの管理

```
mysql> PURGE MASTER LOGS TO 'MySQL.000003';
Query OK, 0 rows affected (0.06 sec)

mysql> SHOW MASTER LOGS;
+---------------+-----------+
| Log_name      | File_size |
+---------------+-----------+
| MySQL.000003 |      2315 |
| MySQL.000004 |       628 |
| MySQL.000005 |      1090 |
| MySQL.000006 |       126 |
| MySQL.000007 |       146 |
| MySQL.000008 |       107 |
+---------------+-----------+
6 rows in set (0.00 sec)
```

バイナリログの管理

```
mysql> RESET MASTER;
Query OK, 0 rows affected (0.06 sec)

mysql> SHOW MASTER LOGS;
+---------------+-----------+
| Log_name      | File_size |
+---------------+-----------+
| MySQL.000001 |       107 |
+---------------+-----------+
1 row in set (0.00 sec)
```

補足：ストレージエンジンごとの特性について

- InnoDB
  - 共有テーブルスペースファイルとInnoDBログファイルの組で1つの単位。**innodb_file_per_table**設定時は ***.ibd** も必要
    - 共有テーブルスペースファイル：**ibdata1**、**ibdata2**、…
    - InnoDBデータファイル：**.ibd**ファイル
    - InnoDBログファイル：**ib_logfile0**、**ib_logfile1**、…
    - テーブル定義ファイル：**.frm**ファイル
  - MySQL5.6ではトランスポータブル表領域機能を利用して、テーブル単位でファイルコピーして移動可能
    - MySQL 5.5以前のバージョンではファイルコピーだけでは移動は出来ない
  - クラッシュセーフ、トランザクション対応
  - ロックは行単位
  - MySQL Enterprise Backup(旧名称 InnoDB Hot Backup)でホットバックアップ可能

補足:ストレージエンジンごとの特性について

- MyISAM
  - 3つのファイルから構成される(**.frm**、**.MYD**、**.MYI**)
  - ファイルとして書込み禁止(**FLUSH TABLES WITH READ LOCK**)すれば
    ファイルコピーできる
  - クラッシュセーフでは無い
    (MySQLサーバやOSのクラッシュ時には**repair**が必要)
  - ロックはテーブル単位

Program Agenda

1. バックアップ手法の概要
2. バックアップ対象のファイル
3. **バックアップ/リストア例**
4. MySQL Enterprise Backup

前提

- 設定ファイル my.cnf の配置先/usr/local/mysql/data/my.cnf

```
[mysqld]
datadir=/usr/local/mysql/data
socket=/usr/local/mysql/data/mysql.sock
user=mysql
log-bin=MySQL
```

OSコマンドによる物理バックアップ例

1. MySQLサーバを停止
   - `$ mysqladmin --user=root --password=root ¥`
     `--socket=/usr/local/mysql/data/mysql.sock shutdown`

2. コールドバックアップを取得する
   - `$ cp -rp /usr/local/mysql/data /backup/mysql-20121121`

3. MySQLサーバを起動する
   - `$ mysqld_safe --defaults-file=/usr/local/mysql/data/my.cnf &`

物理バックアップのリストア例

1. マシンやディスクが壊れている場合は復旧する
2. MySQLサーバが起動している場合は停止する
3. 既存のデータベース領域を削除する
   - `$ rm -rf /usr/local/mysql/data`
4. バックアップファイルをリストアする
   - `$  cp -rp /backup/mysql-20121121 /usr/local/mysql/data`
5. MySQLサーバを起動する
   - `$ mysqld_safe --defaults-file=/usr/local/mysql/data/my.cnf &`

mysqldumpによるバックアップ例

- 全てのテーブルをロックしてデータベース全体のバックアップを取得
  - `$ mysqldump --user=root --password=root --master-data=2 ¥`
    `--socket=/usr/local/mysql/data/mysql.sock ¥`
    `--hex-blob --default-character-set=utf8 --all-databases ¥`
    `--lock-all-tables > mysql_bkup_dump.sql`
- InnoDBのトランザクションを使用してデータベース全体のバックアップを取得
  - `$ mysqldump --user=root --password=root --master-data=2 ¥`
    `--socket=/usr/local/mysql/data/mysql.sock ¥`
    `--hex-blob --default-character-set=utf8 --all-databases ¥`
    `--single-transaction > mysql_bkup_dump.sql`

mysqldumpのオプション

- **--master-data=2**
  - バックアップ取得のバイナリファイル名とバイナリファイル内の位置(**Position**)を
    コメントとしてバックアップファイルに記録
- **--hex-blog**
  - バイナリ型(**BINARY**、**VARBINARY**、**BLOG**) と**BIT**型のデータを16進数表記で出力
- **--default-character-set**
  - **mysqldump**がデフォルトで利用するキャラクタセットを指定。
    通常はMySQLサーバのシステム変数**default-character-set**と同じものを指定すれば良い
- **--all-databases**
  - 全てのデータベースをバックアップ
- **--lock-all-tables**
  - 全てのテーブルをロックしてバックアップを取得する
- **--single-transaction**
  - InnoDBがサポートしているトランザクションの仕組みを利用して、
    InnoDBテーブルに限り一貫性のとれたバックアップを取得する

注意事項：mysqldumpによるバックアップ

- データの整合性を保つために、バックアップ取得中は、テーブルに関するDDL文（※）を実行しないこと

  ※ALTER TABLE, CREATE TABLE, DROP TABLE,RENAME TABLE, TRUNCATE TABLE

- マニュアルの“--single-transaction”オプションの説明部分より引用
  - 「While a --single-transaction dump is in process, to ensure a valid dump file (correct table contents and binary log coordinates), no other connection should use the following statements: ALTER TABLE, CREATE TABLE, DROP TABLE, RENAME TABLE, TRUNCATE TABLE.」

mysqldumpからのリストア例

1. マシンやディスクが壊れている場合は復旧する
2. MySQLサーバが起動している場合は停止する
3. 既存のデータベース領域を削除後、再作成する
   (必要に応じて事前にバックアップを取得)
   - `$  rm -rf /usr/local/mysql/data`
   - `$  mkdir /usr/local/mysql/data`

4. バイナリログの出力を停止 (**my.cnf**から**log-bin**をコメントアウト)
   - ※この例の場合は、バックアップしておいた **my.cnf** を
     /usr/local/mysql/data 配下に配置してから、**log-bin**をコメントアウト

mysqldumpからのリストア例

5. 権限テーブルとネットワーク接続を無効化した状態で
   MySQLサーバを起動
   - `$ mysqld_safe --defaults-file=/usr/local/mysql/data/my.cnf ¥`
     `--skip-networking --skip-grant-tables &`
6. バックアップファイルに記述されたSQL文を実行
   - `$ mysql --default-character-set=utf8 ¥`
     `--socket=/usr/local/mysql/data/mysql.sock < mysql_bkup_dump.sql`
7. バイナリログの出力を再開して、正常に再起動
   - `$ mysqladmin --user=root --password=root ¥`
     `--socket=/usr/local/mysql/data/mysql.sock shutdown`
   - `$ mysqld_safe --defaults-file=/usr/local/mysql/data/my.cnf &`

mysqldumpからのリストアの注意点

- ※MySQL 5.5以上の場合、この手順では性能統計情報を格納する **performance_schema** が作成されず、エラーログに **performance_schema** が無いというエラーが出力される。

- [回避策]

1. サーバ再起動前に、**performance_schema**ディレクトリを **datadir** 配下にコピーする (インストールディレクトリ/**data/performance_schema**)
2. MySQLサーバ再起動後、**mysql_upgrede**を実行する
   - `$ mysql_upgrade  --user=root --password=root ¥`
     `--socket=/usr/local/mysql/data/mysql.sock`

バイナリログを使用したポイントインタイムリカバリ

1. ネットワーク接続を無効化した状態でMySQLサーバを起動
   - `$ mysqld_safe --defaults-file=/usr/local/mysql/data/my.cnf ¥`
     `--skip-networking &`

2. バイナリログからロールフォワード用SQL文を生成し、生成したSQL文を実行する(ファイル名とポジションは、適切なものを指定)
   - `$ mysqlbinlog --disable-log-bin --start-position=1017 ¥`
     `MySQL.000010 MySQL.000011 > recover.sql`
   - `$ mysql --user=root --password=root ¥`
     `--socket=/usr/local/mysql/data/mysql.sock ¥`
     `--default-character-set=utf8 < recover.sql`

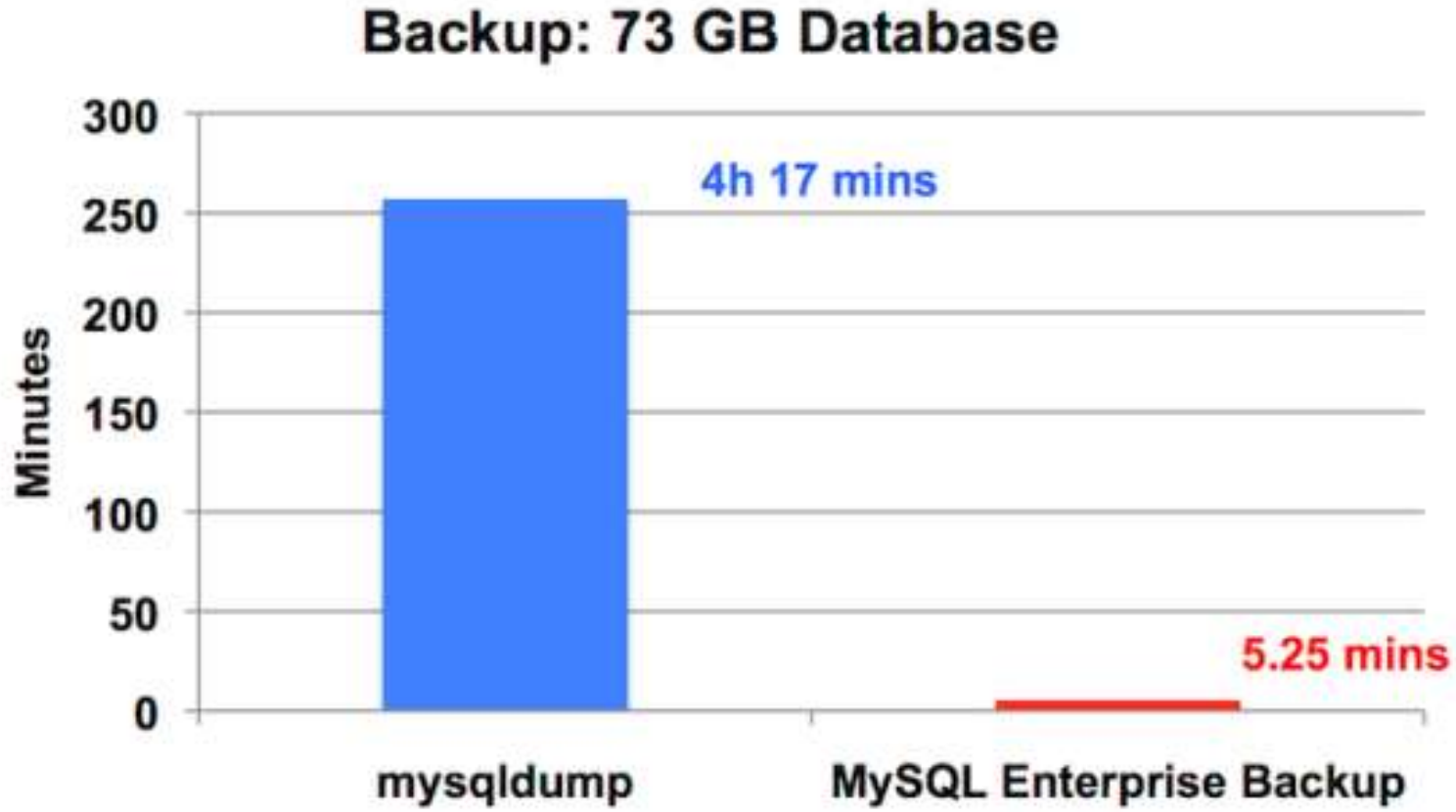
バイナリログを使用したポイントインタイムリカバリ

3. 正常に再起動
   - `$  mysqladmin --user=root --password=root ¥ --socket=/usr/local/mysql/data/mysql.sock shutdown`
   - `$ mysqld_safe --defaults-file=/usr/local/mysql/data/my.cnf &`

Program Agenda

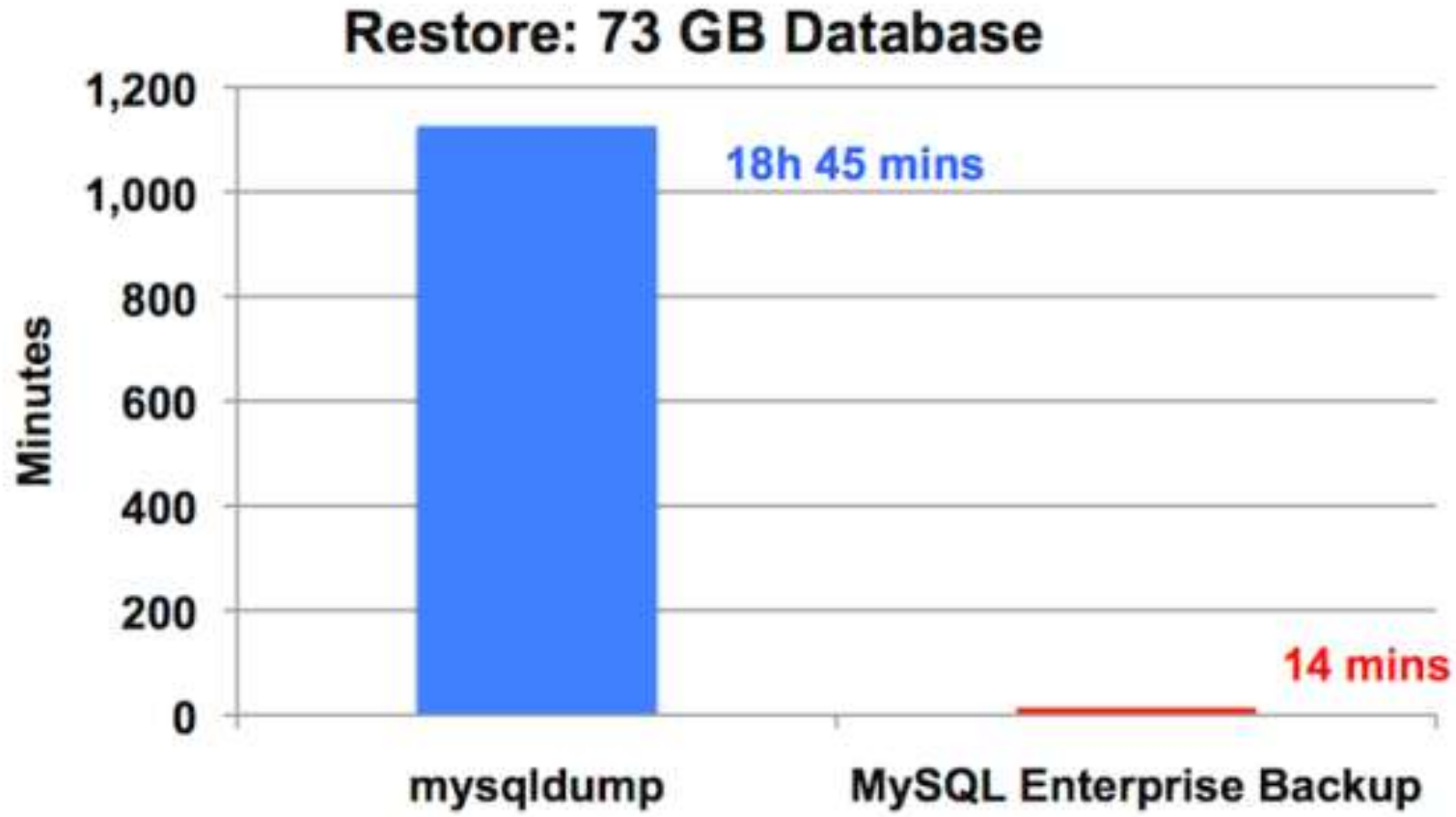
1. バックアップ手法の概要
2. バックアップ対象のファイル
3. バックアップ/リストア例
4. MySQL Enterprise Backup

高速なバックアップ



mysqldumpより49倍速い

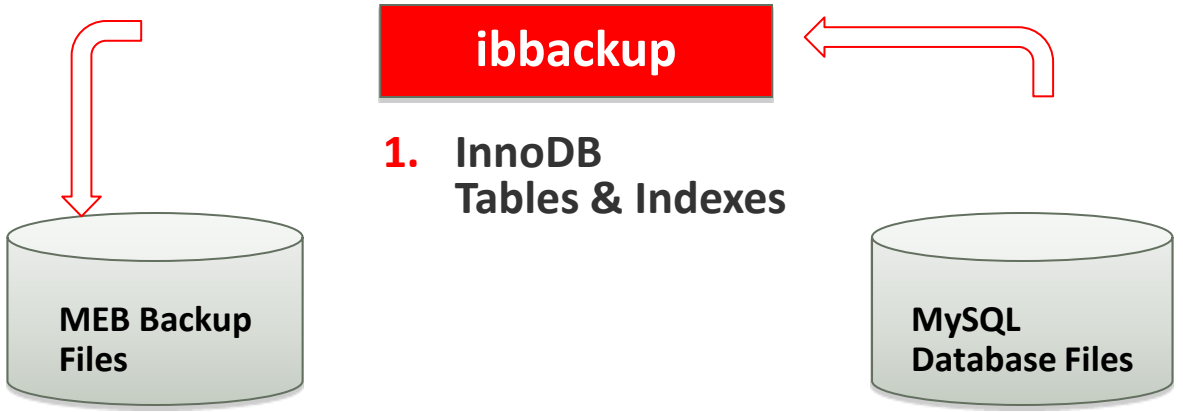
高速なリストア



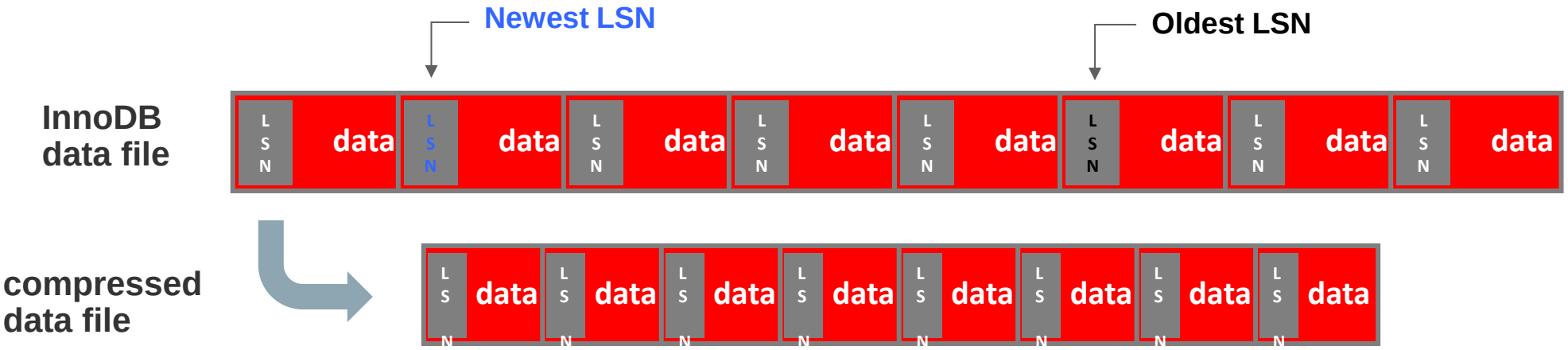
mysqldumpより80倍速い

MySQL Enterprise Backupの動作

- Step 1: Backing Up InnoDB Data Files
  - Copies and compresses InnoDB data files
  - System Database (ibdata) & Single-table Tablespaces (.ibd)
- Produces “Fuzzy Backup
  - Backup of data files doesn’t correspond to any specific log sequence number (LSN)
  - Different database pages are copied at varying times



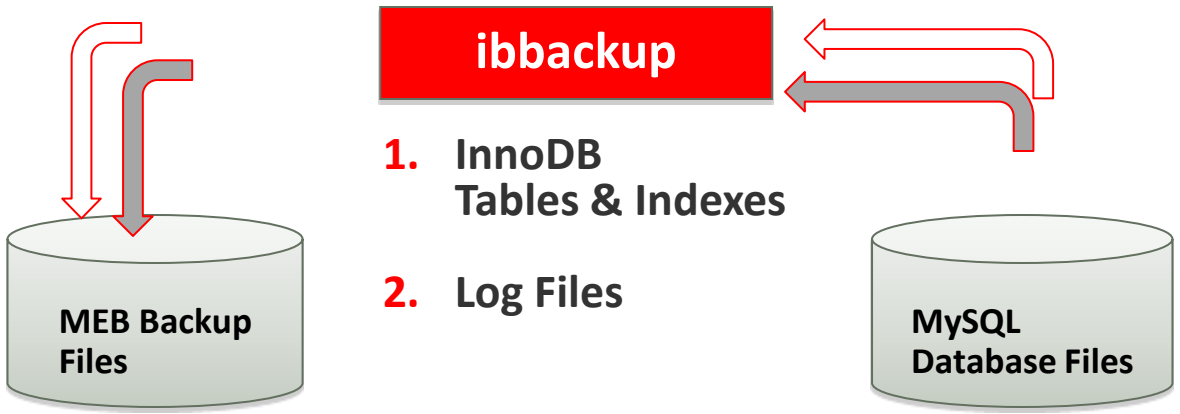
Backing Up InnoDB Data Files



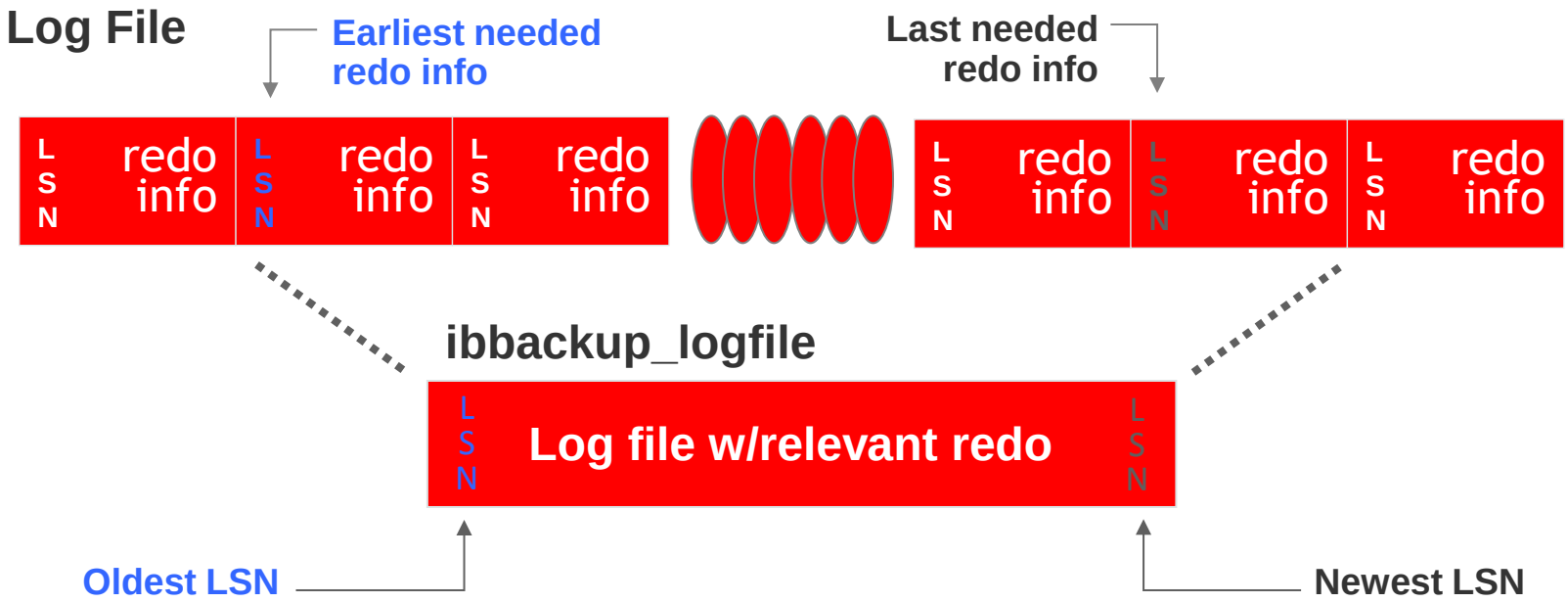
- Backup files size is reduced by 70%
  - Omits unused storage in each block, empty pages
- Produces “Fuzzy Backup”
- Notes earliest and latest Log Sequence Number (LSN)

MySQL Enterprise Backupの動作

- Step 2: Backing up InnoDB Log Files
  - Copies Log Records accumulated during data file copy
  - All redo records with LSNs during data file copy



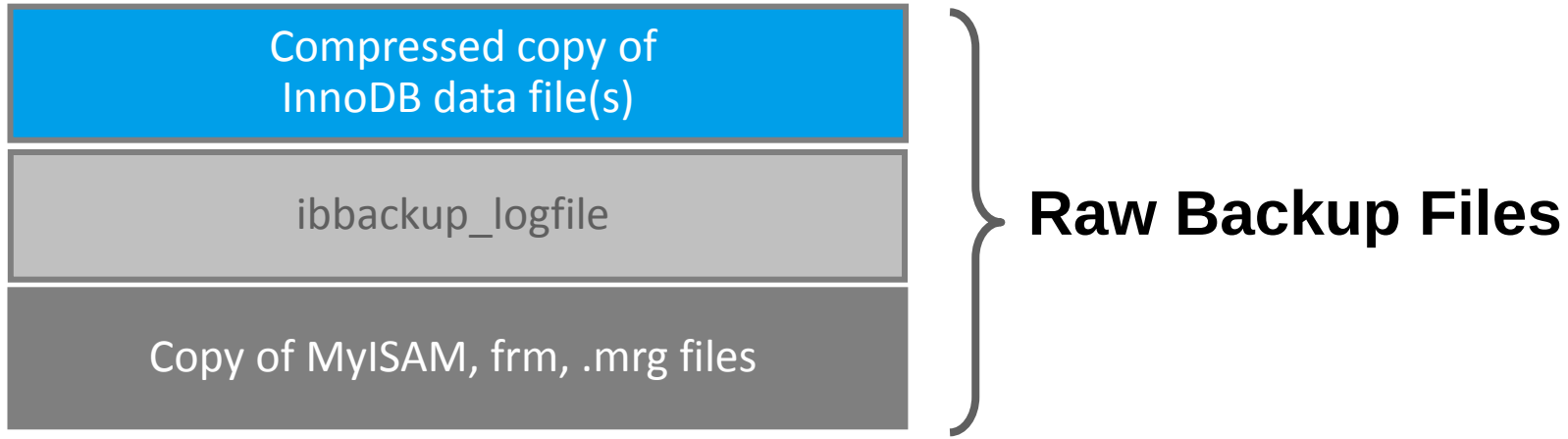
Backing up InnoDB Log Files



- Copies portion of the log file that contains all required redo information
- Covers the time from beginning to end of data backup
- Recovers all data blocks modified after copied to compressed data file

Tips: “Raw Backup” Files

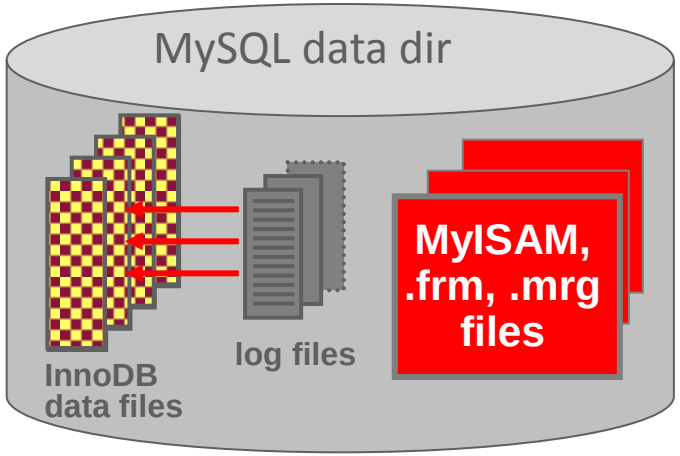
- The “raw backup” files from backup phase cannot be directly consumed by MySQL
- These files can be copied to media
- The database must be “restored” first
- Use mysqlbackup to restore database before use



Restoring a Database

Compressed copy of InnoDB data file(s)

1. **Uncompresses InnoDB files to data dir**
2. **Recreates InnoDB Log files**
3. **Applies log, so InnoDB files are consistent**
4. **Restores MyISAM and other files**



ibbackup_logfile

Copy of MyISAM, frm, .mrg files

バックアップオプション

- 圧縮バックアップ : --compress, --compress-level=LEVEL
- SSH経由のストリーム・バックアップ
- テープデバイスへの直接バックアップ、SBTインターフェース利用
- 部分バックアップ: --include, --with-tts
- 増分バックアップ
- 未使用領域のスキップ: --skip-unused-pages
- クラウドストレージとの直接連携(S3, etc)
- 暗号化(AES256)

バックアップ操作(基本)

- バックアップ取得とログ適用を1回の操作で実行

```
shell# mysqlbackup --user=root -p --backup-dir=/home/admin/backups ¥
> backup-and-apply-log
```

- バックアップ取得とログ適用を分けて実行

```
shell# mysqlbackup --user=root -p --backup-dir=/home/admin/backups ¥
> backup

shell# mysqlbackup --user=root -p --backup-dir=/home/admin/backups ¥
> apply-log
```

リストア操作(基本)

- ログ適用済みのバックアップをリストア

```
shell# mysqlbackup --defaults-file=/usr/local/mysql/my.cnf ¥
> --backup-dir=/home/admin/backups ¥
> copy-back
```

- ログ適用前のバックアップにログを適用し、リストア

```
shell# mysqlbackup --defaults-file=/usr/local/mysql/my.cnf ¥
> --backup-dir=/export/backups/full ¥
> copy-back-and-apply-log
```

バックアップオプション：圧縮

- データによっては80%以上の大幅な圧縮も可能

```
shell# mysqlbackup --backup-dir=/opt/mysql/backups/uncompressed ¥
> backup 2>/dev/null && du -csh /opt/mysql/backups/uncompressed
334M    /opt/mysql/backups/uncompressed
334M    total


shell# mysqlbackup --backup-dir=/opt/mysql/backups/compressed ¥
> --compress --compress-level=9 backup 2>/dev/null && du -csh ¥
> /opt/mysql/backups/compressed
81M     /opt/mysql/backups/compressed
81M     total
```

バックアップオプション：ストリーム・バックアップ

- リモートサーバにバックアップデータを出力
- テンポラリディレクトリを一時出力先として指定

```
shell# mysqlbackup --backup-image=- --backup-dir=/tmp/backup ¥
> --disable-manifest backup-to-image 2>/dev/null | ssh matt@solo ¥
> "cat > /tmp/backup.img“


shell# ssh matt@solo "ls -lh /tmp/backup.img"
-rw-r--r--   1 matt     staff       333M Sep 10 15:54 /tmp/backup.img
```

バックアップオプション：SBTインターフェースの利用

- 既存のシステムバックアップツールとの統合可能
  - Oracle Secure Backup (fully supported)
  - IBM Tivoli Storage Manager
  - Symantec NetBackup

```
shell-osb# mysqlbackup --backup-image=sbt:backup-mattprod-2013-09-08 ¥
> --backup-dir=/tmp/backup backup-to-image


shell-tsm# mysqlbackup --backup-image=sbt:my-tsm-backup ¥
> --sbt-lib-path=/usr/lib/libobk.so ¥
> --sbt-environment="TDPO_OPTFILE=/opt/ibm/tsm/tdpo.opt" ¥
> --backup-dir=/tmp/backup backup-to-image
```

バックアップオプション：部分バックアップ

- 重要なテーブル/スキーマのみをバックアップ
- InnoDBのトランスポータブル・テーブルスペース(tss)もサポート

```
shell# mysqlbackup --include=sakila.* --only-innodb-with-frm=related ¥
> backup


shell# mysqlbackup --use-tts=with-minimum-locking ¥
> --include=employees.* --backup-dir=/opt/mysql/backups backup
```

バックアップオプション：増分バックアップ

- 前回のフルバックアップからの変更点のみをバックアップ

```
shell# mysqlbackup --incremental ¥
> --incremental-base=dir:/opt/mysql/backup/monday ¥
> --incremental-backup-dir=/opt/mysql/backup/tuesday backup


shell# mysqlbackup --incremental ¥
> --incremental-base=history:last_backup --with-timestamp ¥
> --incremental-backup-dir=/opt/mysql/backup backup
```

バックアップオプション：未使用領域のスキップ

- InnoDBの空白領域や未使用領域はバックアップしない
- このオプションと圧縮を組み合わせてバックアップの効率化

```
shell# mysqlbackup --backup-dir=/opt/mysql/backups/compressed ¥
> --compress --compress-level=9 --skip-unused-pages backup
```

バックアップオプション：クラウドストレージとの直接連携

- Amazon S3上へバックアップを取得
  (MEB3.11時点で対応しているのはAmazon S3のみ)

```
shell# mysqlbackup¥
> --cloud-service=s3 --cloud-aws-region=<aws region> ¥
> --cloud-access-key-id=<aws access key id> ¥
> --cloud-secret-access-key=< aws secret access key> ¥
> --cloud-bucket=<s3 bucket name> --cloud-object-key=<aws object key> ¥
> --backup-dir=/home/user/dba/s3backuptmpdir ¥
> --backup-image=- ¥
> backup-to-image
```

リストアオプション：クラウドストレージとの直接連携

- Amazon S3上のバックアップをリストア
  (MEB3.11時点で対応しているのはAmazon S3のみ)

```
shell# mysqlbackup¥
> --cloud-service=s3 --cloud-aws-region=<aws region> ¥
> --cloud-access-key-id=<aws access key id> ¥
> --cloud-secret-access-key=< aws secret access key> ¥
> --cloud-bucket=<s3 bucket name> --cloud-object-key=<aws object key> ¥
> --backup-dir=/home/user/dba/s3backuptmpdir ¥
> --backup-image=- ¥
> image-to-backup-dir
```

バックアップオプション：暗号化

- 暗号化キーを直接指定する場合

```
shell# mysqlbackup --backup-image=/backups/image.enc --encrypt ¥
> --key=23D987F3A047B475C900127148F9E039485798364519287 4A2B3049570C12A34 ¥
> --backup-dir=/var/tmp/backup backup-to-image
```

- ファイルに格納した暗号化キーを使用する場合

```
shell# mysqlbackup --backup-image=/backups/image.enc --encrypt ¥
> --key-file=/meb/key --backup-dir=/var/tmp/backup backup-to-image
```

バックアップオプション：復号化

- ファイルに格納した復号化キーを直接指定する場合

```
shell# mysqlbackup --backup-image=/backups/image.enc --decrypt ¥
> --key-file=/meb/key --backup-dir=/backups/extract-dir extract
```

Hardware and Software
Engineered to Work Together


ORACLE®