

ORACLE®

ORACLE®

MySQL 5.6 レプリケーションとGTID

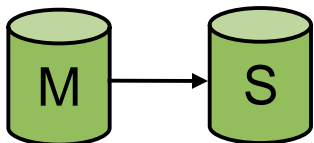
MySQL Global Business Unit
Sales Consulting Manager, JAPAC
梶山 隆輔 / Ryusuke Kajiyama



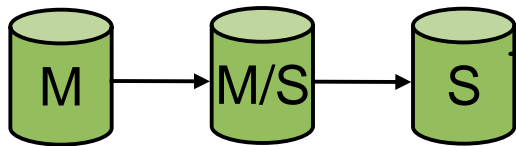
- MySQL レプリケーション
- GTID (Global Transaction Identifiers)
- MySQL Utilities

レプリケーション: マスター→スレーブのデータコピー

- **マスターサーバ**
 - データを変更
 - 変更点をスレーブに転送
- **スレーブサーバ**
 - マスターでの変更点を受け取る
 - 変更点をデータベースに反映

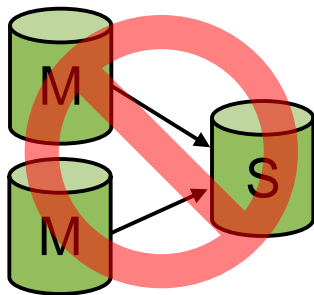
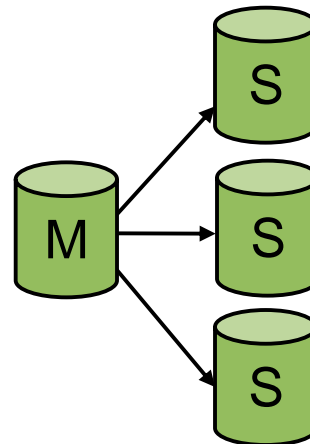


レプリケーション: マスター→スレーブのデータコピー



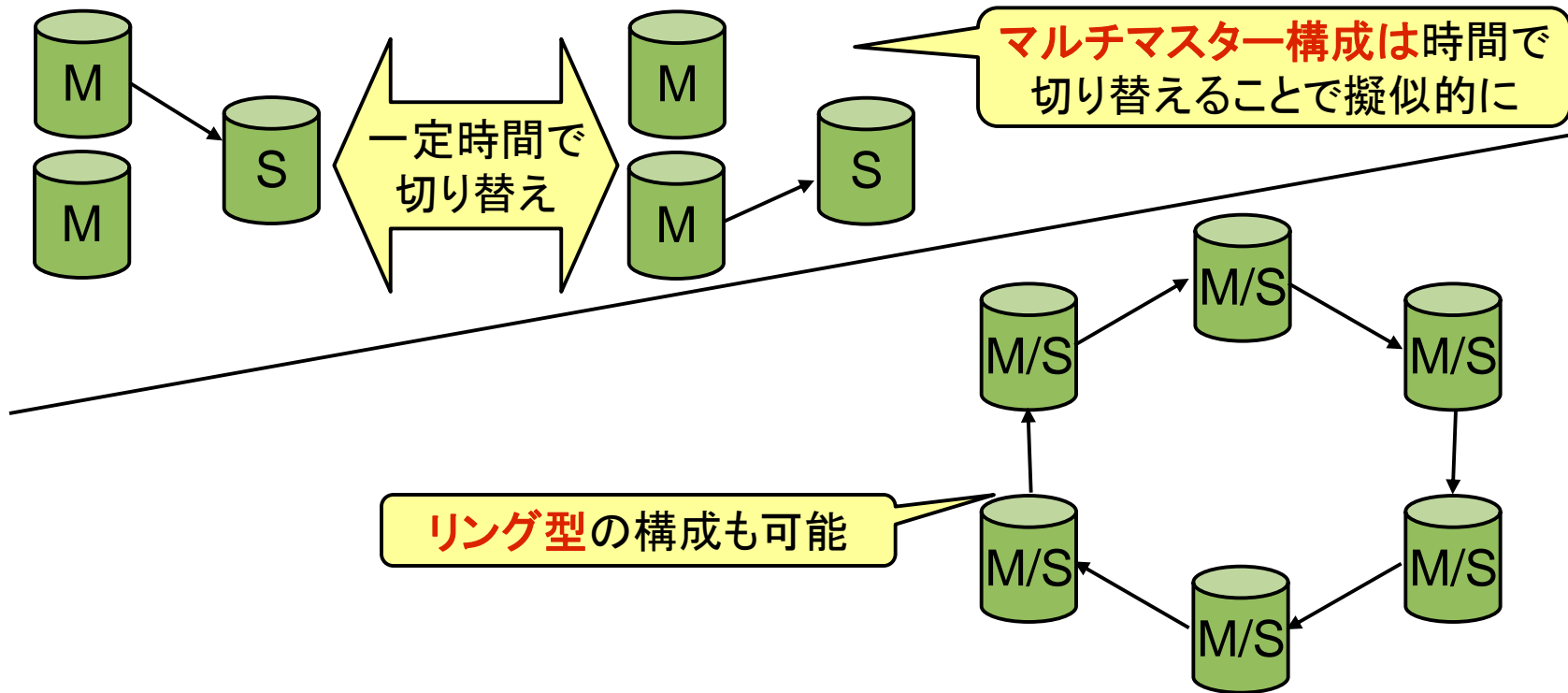
サーバは**マスター、スレーブまたは両方**になれる

マスターは**複数のスレーブ**を持てる



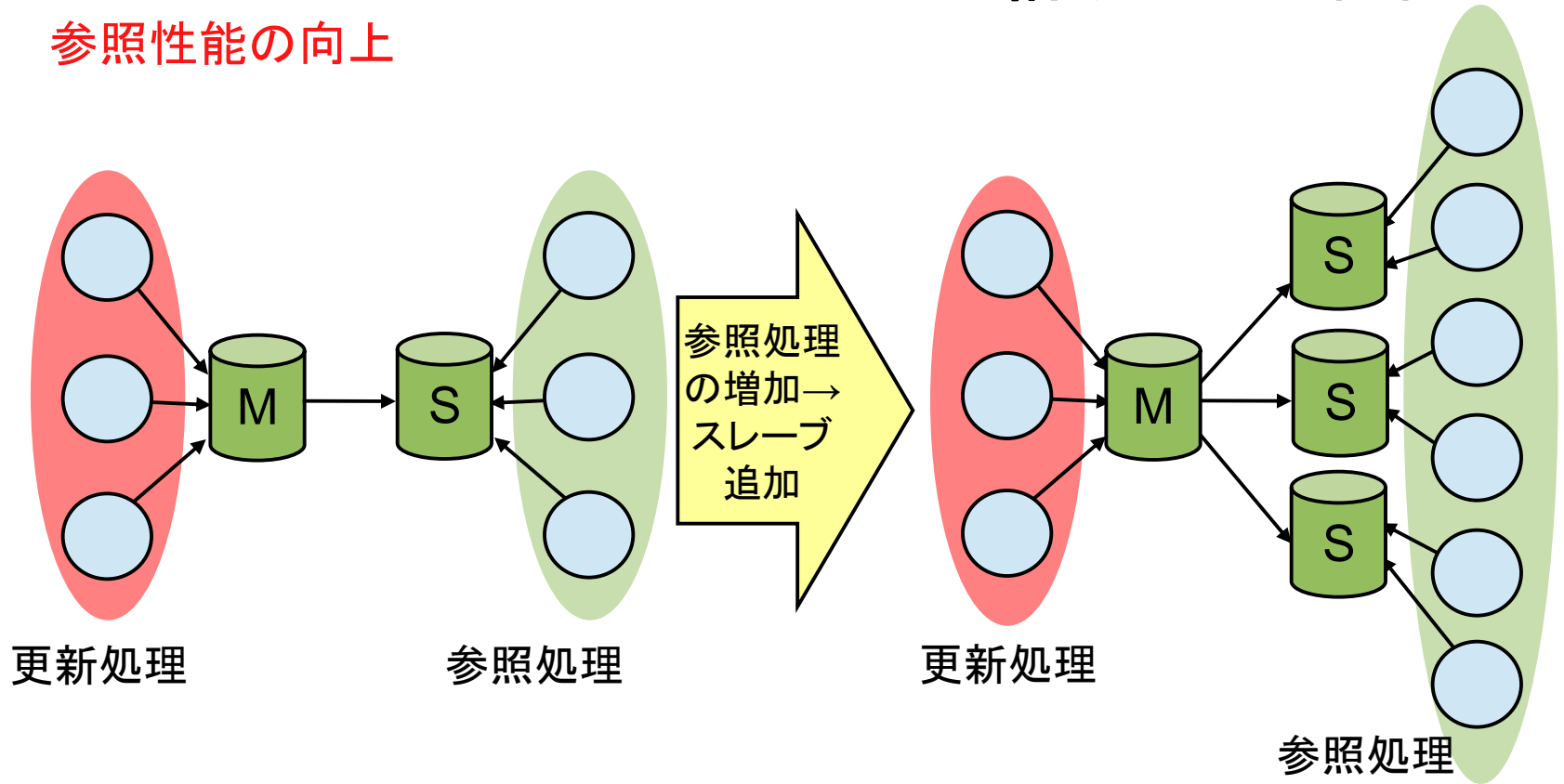
スレーブは**1つのマスターのみ**を持てる

レプリケーション: マスタ→スレーブのデータコピー



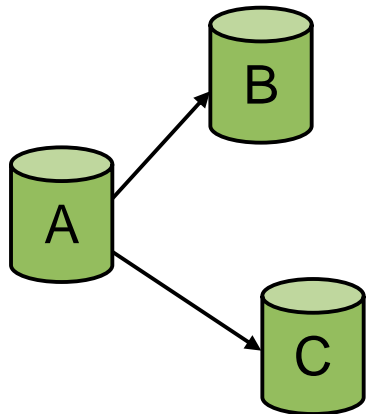
レプリケーション: スケールアウト構成による性能向上

参照性能の向上



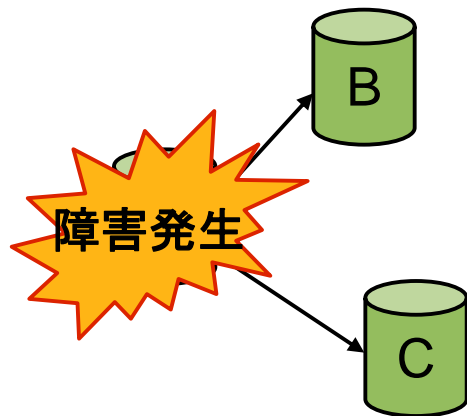
レプリケーション: 高可用性構成

- マスターの障害時に、スレーブをマスターに昇格



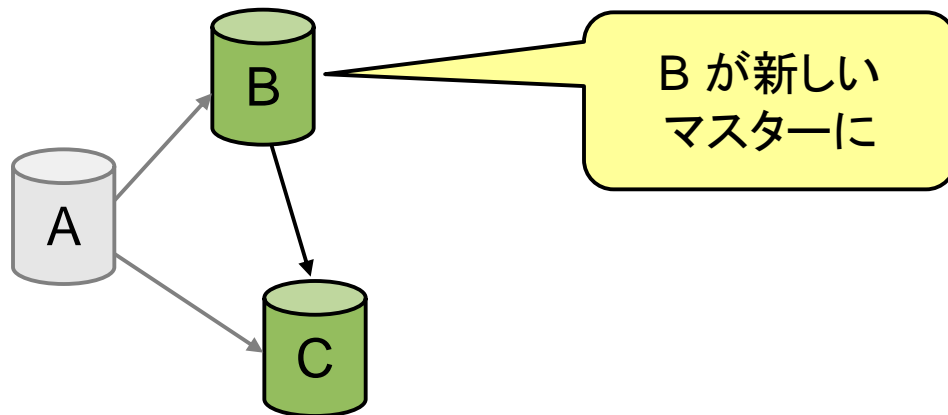
レプリケーション: 高可用性構成

- マスターの障害時に、スレーブをマスターに昇格



レプリケーション: 高可用性構成

- マスターの障害時に、スレーブをマスターに昇格



レプリケーション: 地理的冗長性

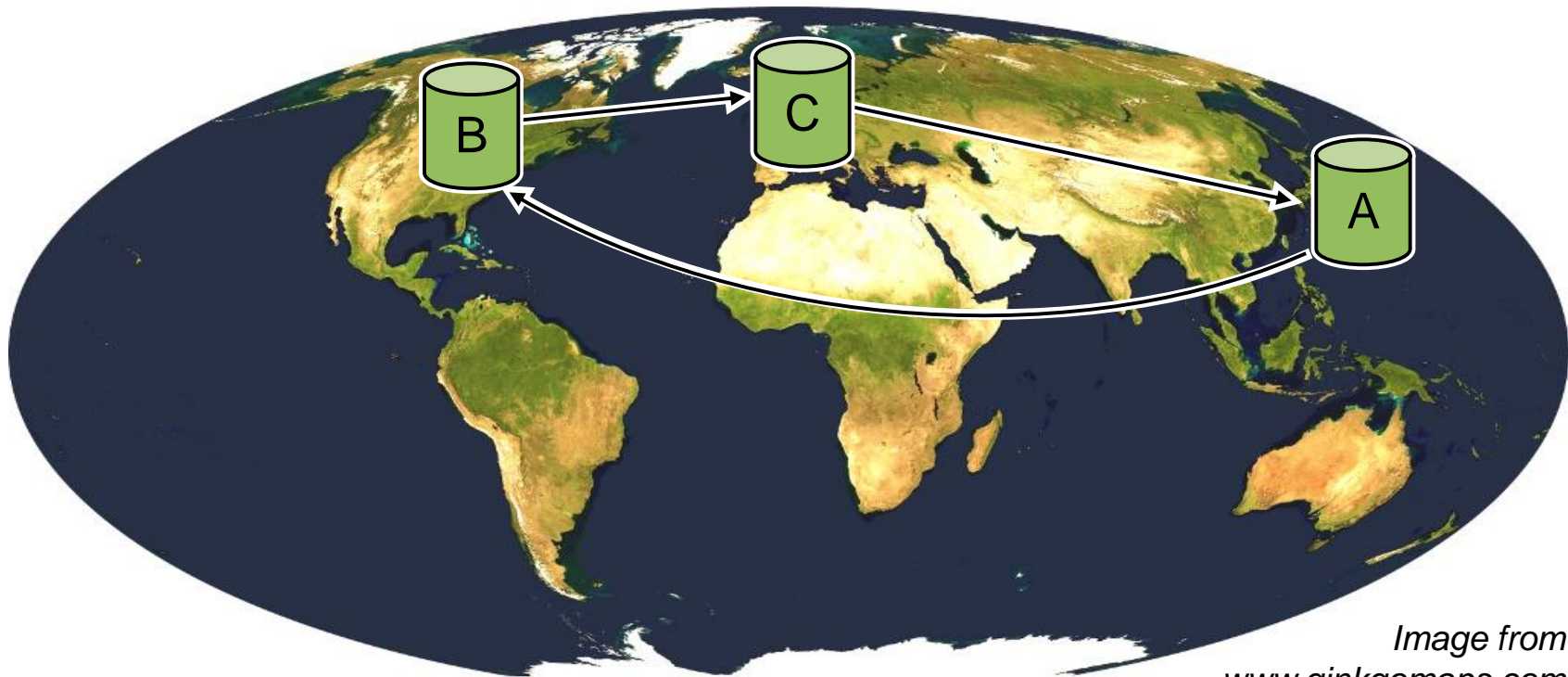
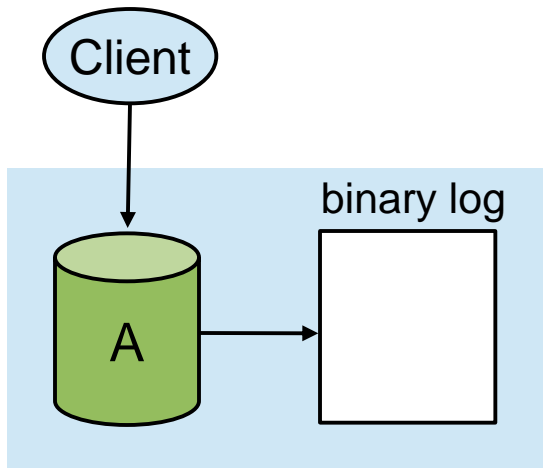
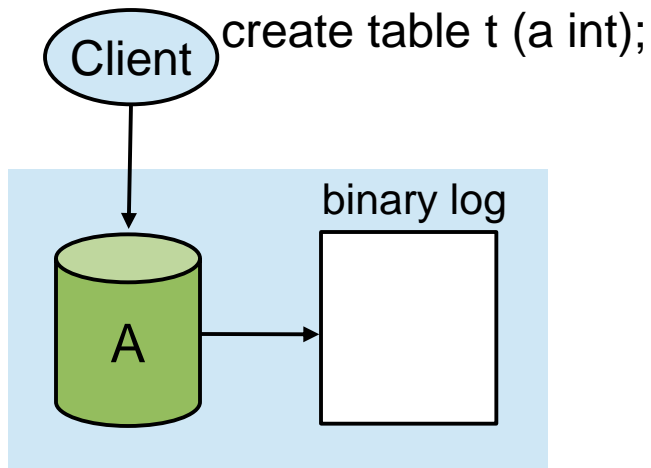


Image from
www.ginkgomaps.com

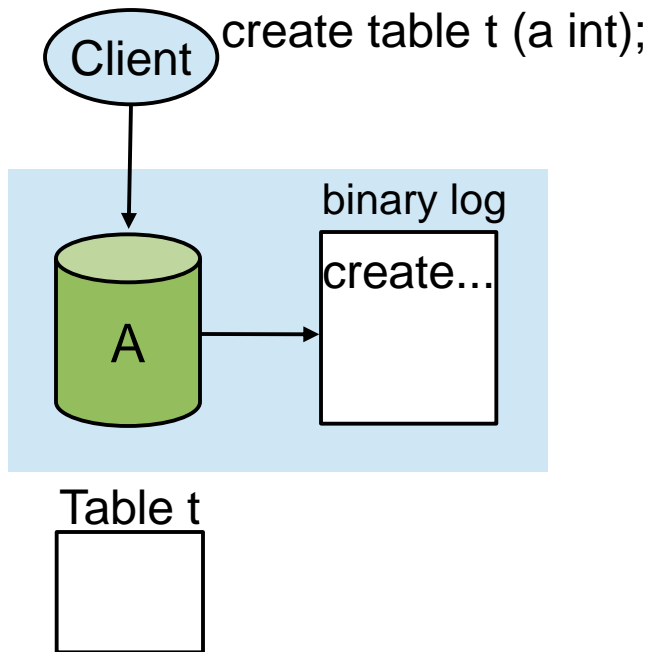
全ての変更点をバイナリログに記録



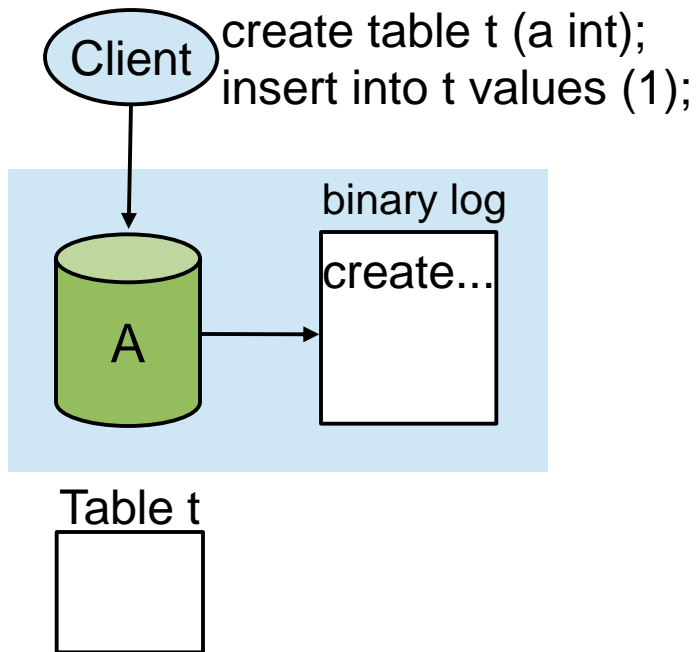
全ての変更点をバイナリログに記録



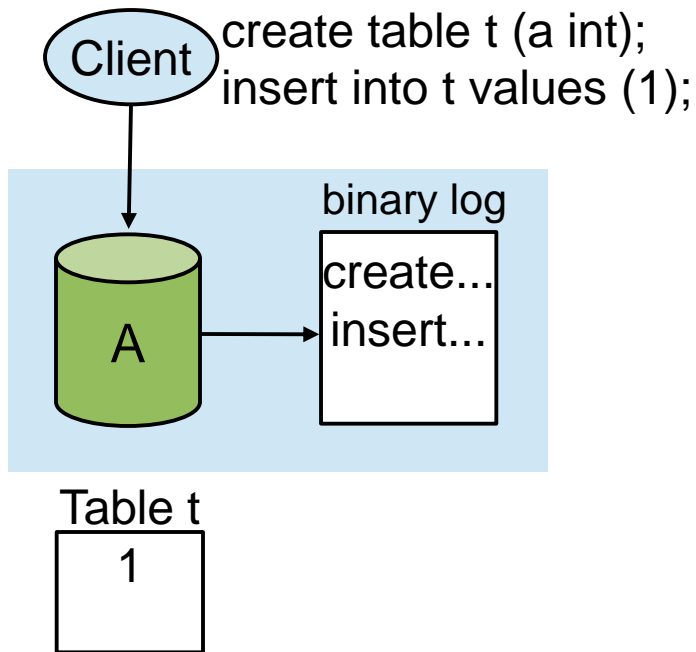
全ての変更点をバイナリログに記録



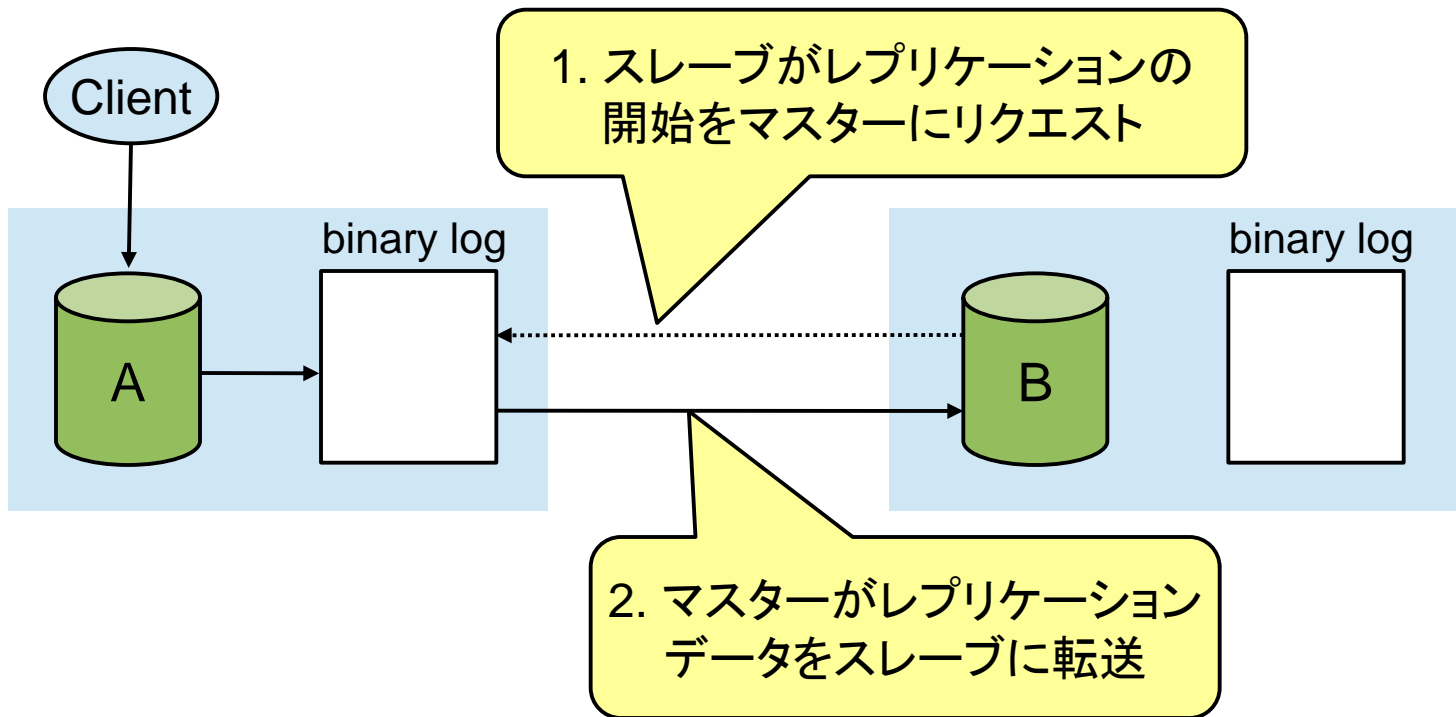
全ての変更点をバイナリログに記録



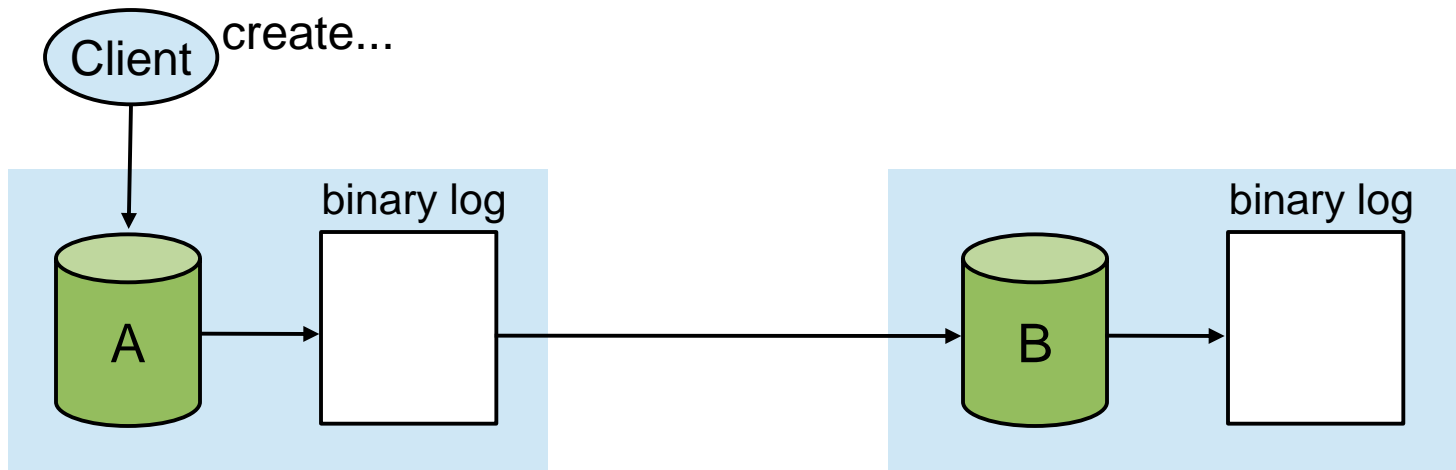
全ての変更点をバイナリログに記録



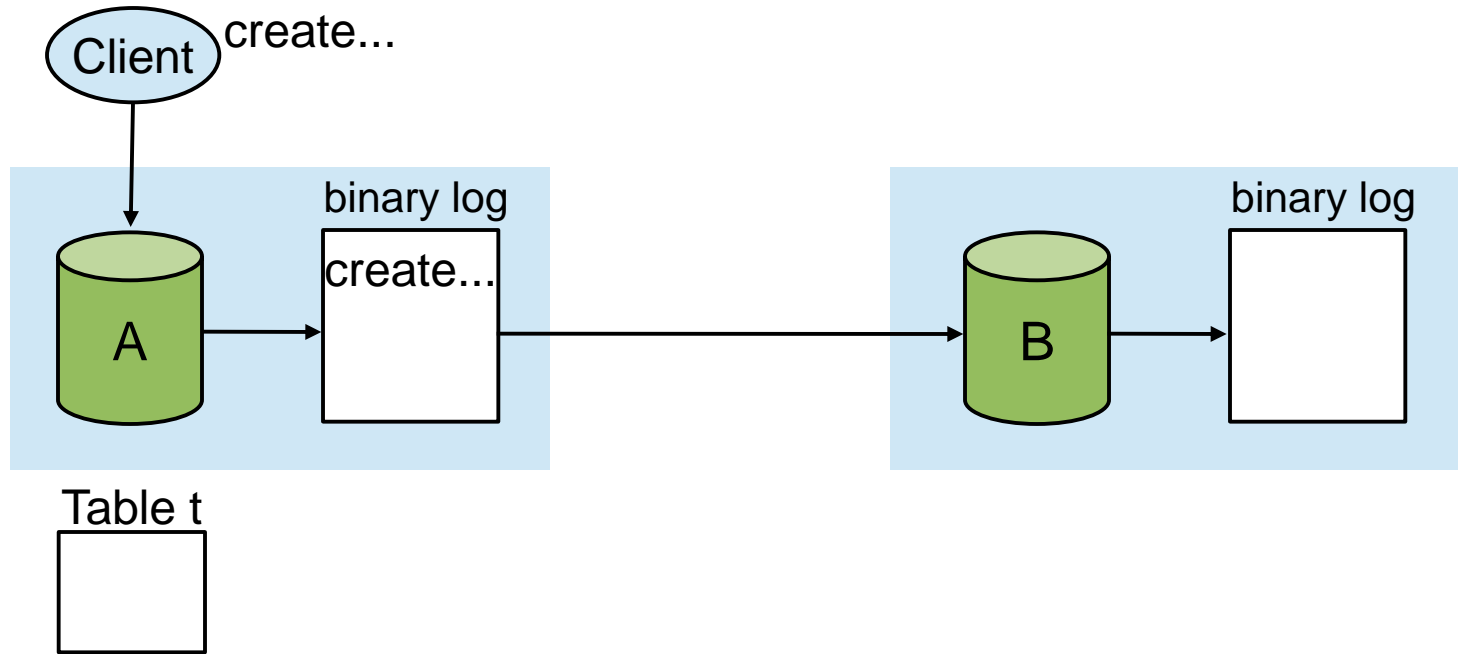
スレーブからレプリケーションを開始



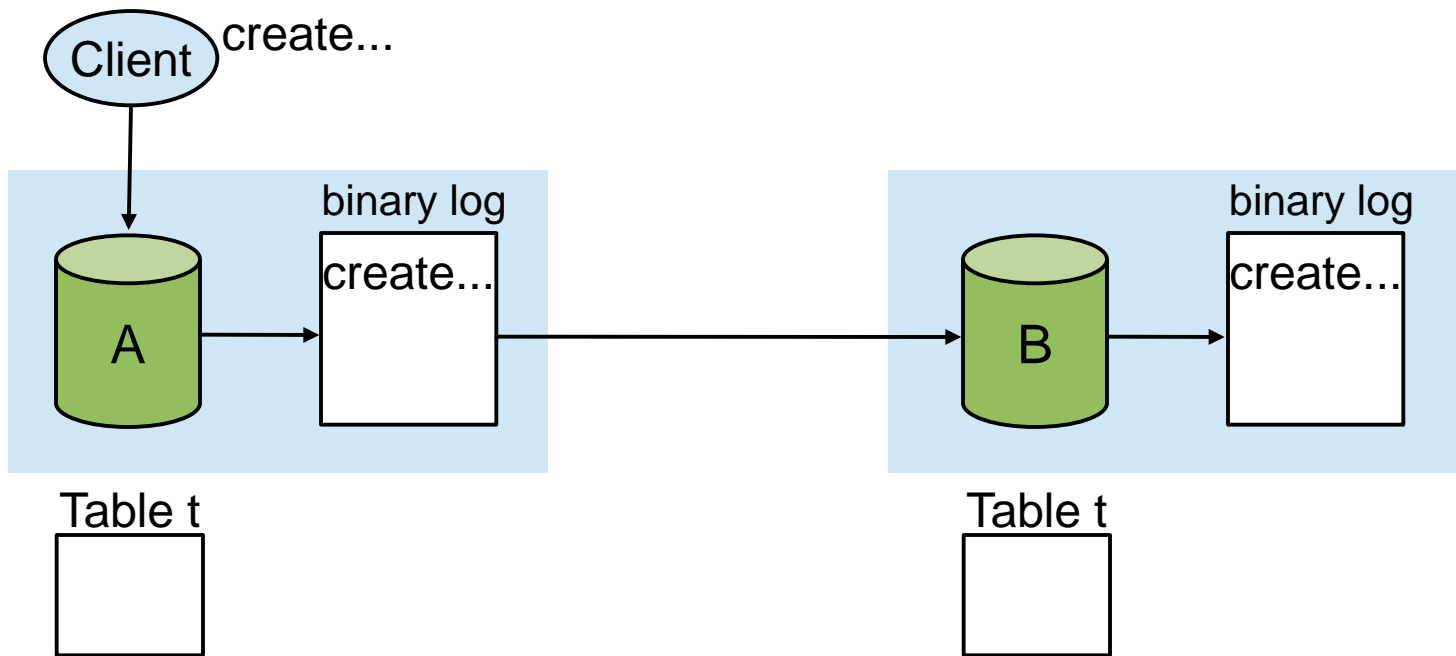
バイナリログの内容をスレーブに転送し、実行



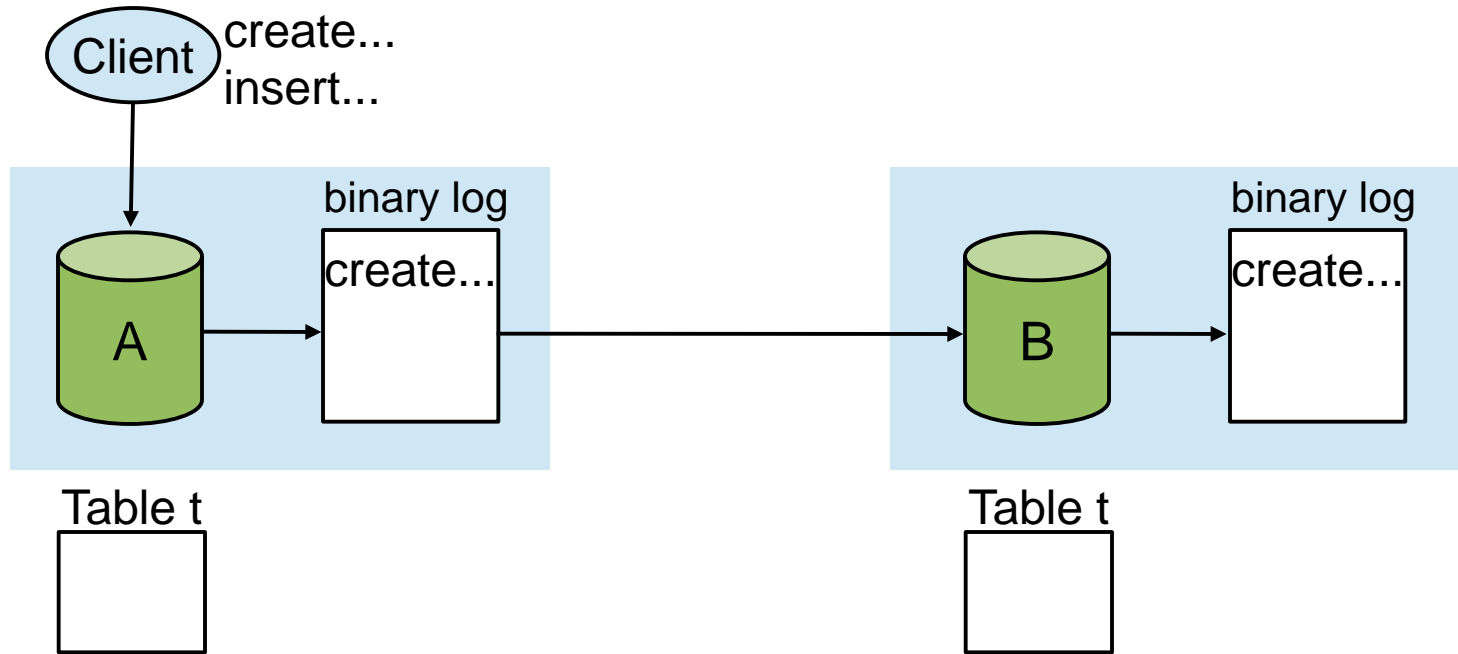
バイナリログの内容をスレーブに転送し、実行



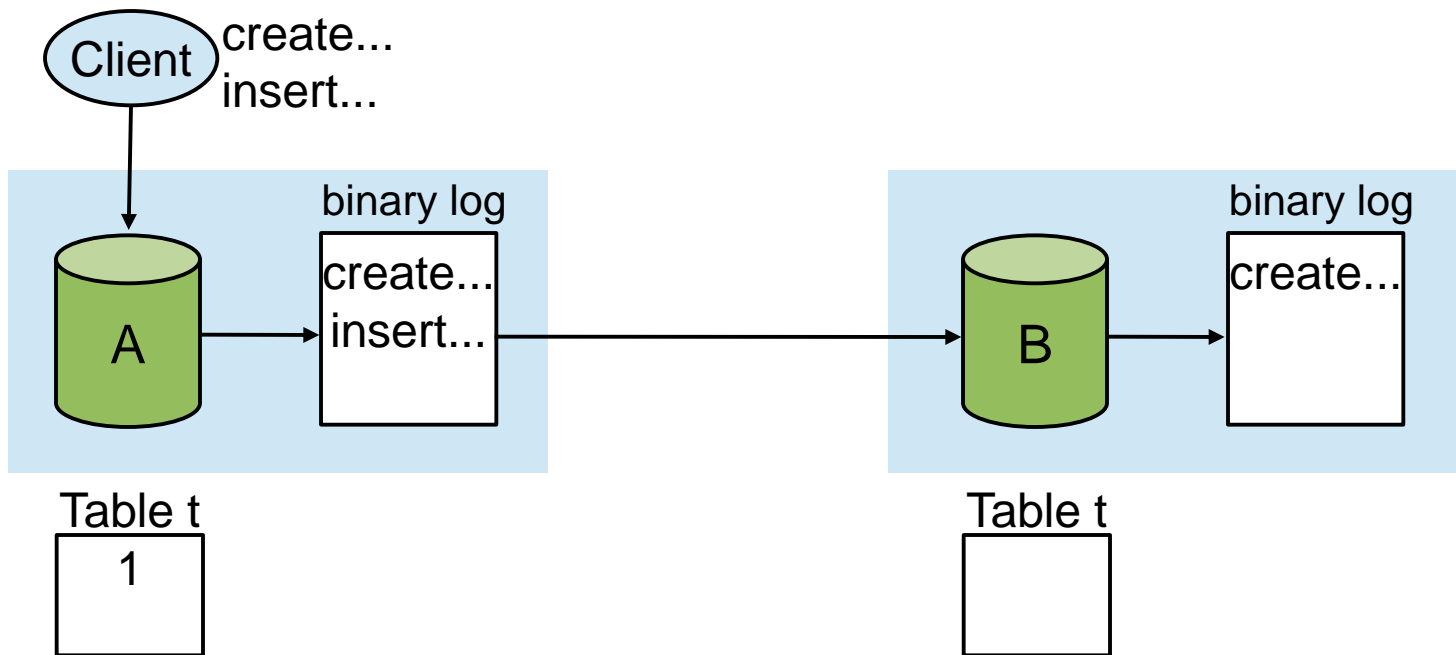
バイナリログの内容をスレーブに転送し、実行



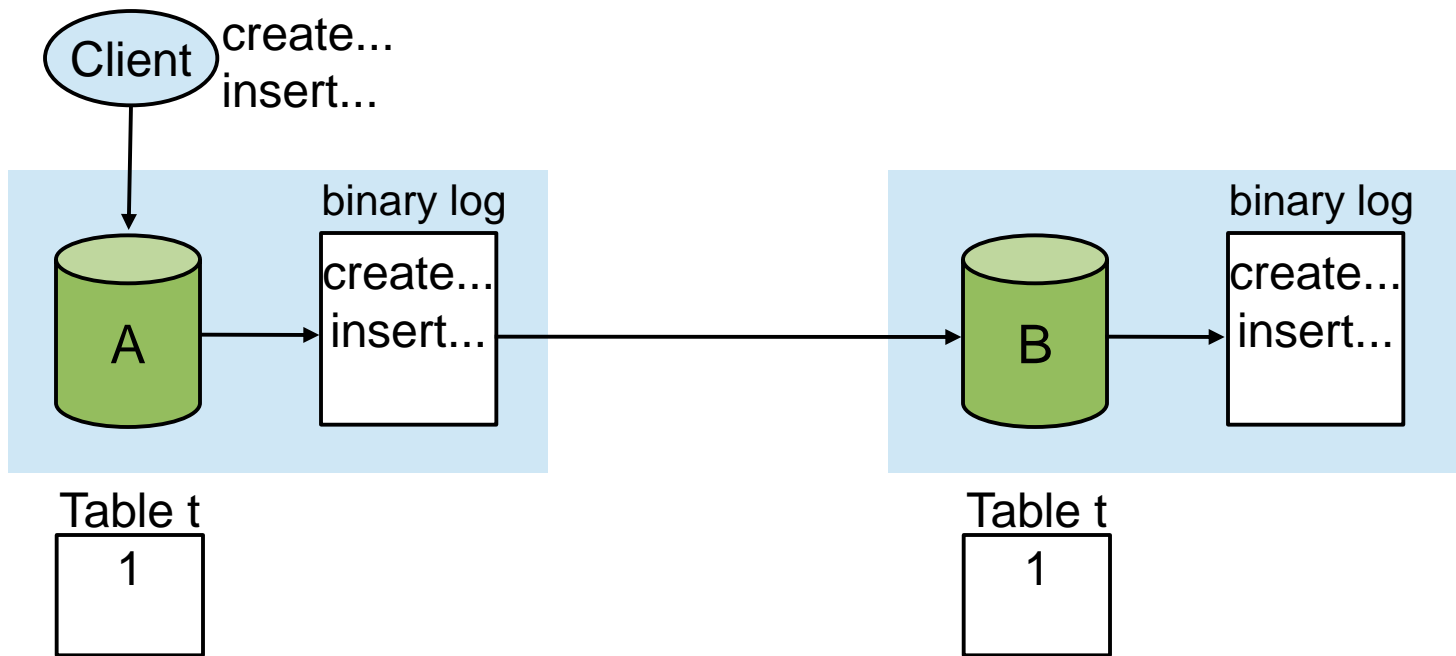
バイナリログの内容をスレーブに転送し、実行



バイナリログの内容をスレーブに転送し、実行



バイナリログの内容をスレーブに転送し、実行



GTID

(Global Transaction Identifiers)

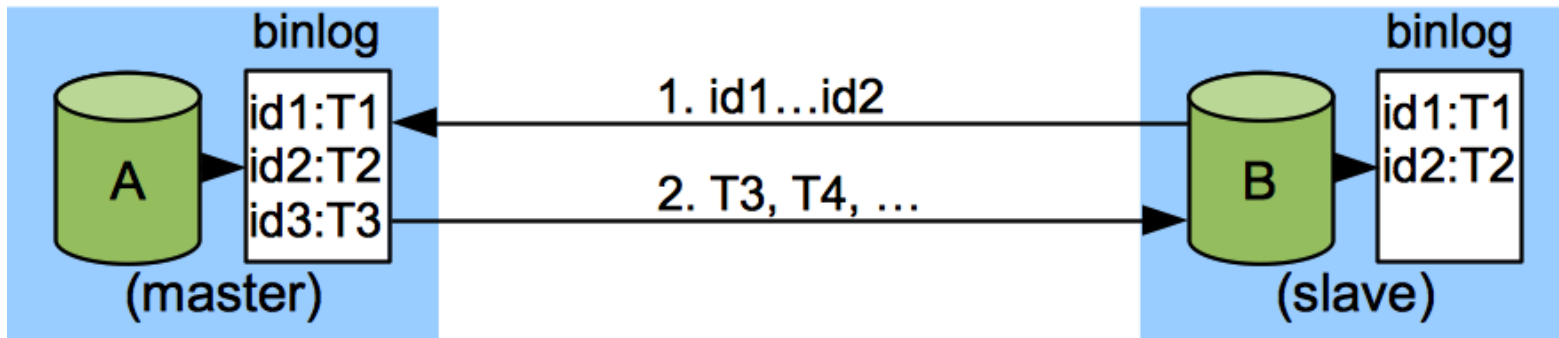
Global Transaction Identifiers

- コミットごとにトランザクションを一意に識別するIDを生成:
 - **server_uuid:number**
a61678ba-4889-4279-9e58-45ba840af334:1
 - **server_uuid** サーバを識別するユニークなID
 - **number** はトランザクション実行ごとに1ずつカウントアップ
- GTIDはバイナリログに記録される
- トランザクションをスレーブで実行する際は同じGTIDが使われる



Global Transaction Identifiers

- 新しいレプリケーションの protocol:
 - スレーブは *master:range* を表す ID をマスターに送る
 - マスタは *range* 以降の全てのトランザクションをスレーブに送る



Global Transaction Identifiers

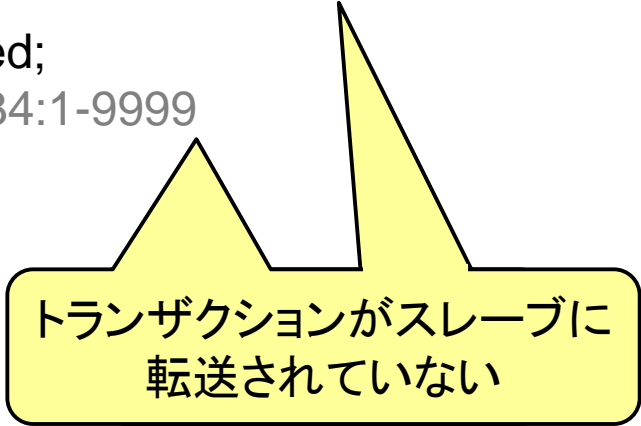
```
master> CREATE TABLE t1 (a INT);
master> SELECT @@global.gtid_executed;
a61678ba-4889-4279-9e58-45ba840af334:1
master> INSERT INTO t1 VALUES (1);
master> INSERT INTO t1 VALUES (2);
master> SELECT @@global.gtid_executed;
a61678ba-4889-4279-9e58-45ba840af334:1-3
```

新しい変数:
gtid_executed

右側はIDの範囲

Global Transaction Identifiers

```
master> SELECT @@global.gtid_executed;  
a61678ba-4889-4279-9e58-45ba840af334:1-10000  
  
slave> SELECT @@global.gtid_executed;  
a61678ba-4889-4279-9e58-45ba840af334:1-9999
```



トランザクションがスレーブに
転送されていない

ハンズオン

▪ 必要事項:

- トランザクションをサポートしたストレージエンジン(InnoDB)のテーブルとサポートしないストレージエンジン(MyISAMなど)のテーブルを、同一のトランザクション内やSQL文で変更しない (5.6.9以降)
- CREATE TABLE ... SELECT を使用しない
- CREATE TEMPORARY TABLE や DROP TEMPORARY TABLE をトランザクション内部で実行しない

ハンズオン

- レプリケーション開始の準備

1. データの同期を取り、全てのサーバを一旦停止

2. 全てのmy.cnfに下記を追加:

```
gtid-mode=on
enforce-gtid-consistency=on
log-bin
log-slave-updates
```

3. 全てのサーバを起動

4. 実行:

```
> CHANGE MASTER TO MASTER_AUTO_POSITION = 1
```

ハンズオン

- フェールオーバー

slaveにて、新しいマスターを指定:

```
> CHANGE MASTER TO MASTER_HOST = '<host>',  
                    MASTER_PORT = <port number>,  
                    MASTER_USER = '<user name>'  
                    MASTER_PASSWORD = 'secret';
```

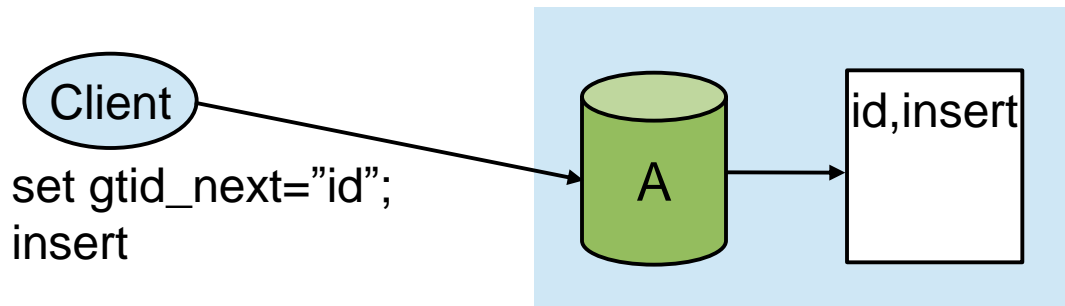
- 「バイナリログポジション」の指定は不要

GTID_NEXT

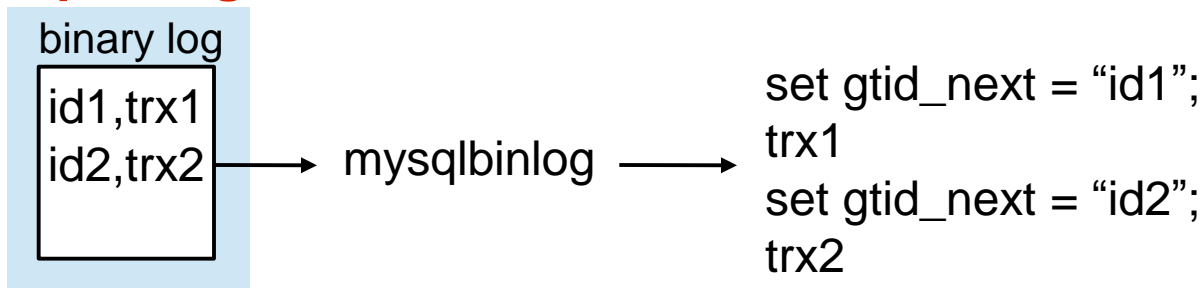
- **GTID_NEXT** – セッションごとのシステム変数
- デフォルト: “**AUTOMATIC**”
 - サーバが次のトランザクション用に **GTID**を生成
- スレーブのスレッドが “**UUID:NUMBER**” を設定
 - サーバは **指定されたGTID** を次のトランザクションで使用

Under the Hood: GTID_NEXT

- **クライアント** もGTID_NEXT を取得可能:



- **mysqlbinlog** が SET GTID_NEXT 文を出力:



MySQL Utilities

MySQL Utilitiesとは？

- MySQLを管理するためのPythonスクリプト集
- 最新バージョンは1.2.0
- MySQL Workbench 5.2.31以降に同梱
 - 現在は5.2.47
- ライセンスはGPLv2
- コードライブラリを用意しているので拡張が容易
- さまざまな運用管理タスクをカバーできるツール作りが目標

データベース管理

- mysqldbcompare – データや定義を比較
- mysqldbcopy – 別のサーバにデータベースをコピー
- mysqldbexport – データとメタデータをエクスポート
- mysqldbimport – データとメタデータをインポート
- mysqldiff – サーバ間のテーブルなどオブジェクトの定義を比較

各種操作

- mysqldiskusage – データベースおよびデータファイルのサイズを表示
- mysqlindexcheck – インデックスの重複をチェック
- mysqlmetagrep – テーブル定義のメタデータをgrep (正規表現利用可)
- mysqlprocgrep – プロセス情報をgrep (正規表現利用可)
- mysqluserclone – 別のサーバにユーザアカウントをコピー
- mysqluc – コマンドライン環境

New!

高可用性関連

- `mysqlfailover` – レプリケーションの自動フェールオーバー
- `mysqlreplicate` – レプリケーションを設定
- `mysqlrpladmin` – レプリケーションの各種管理
- `mysqlrplcheck` – レプリケーションが正しく設定されているかの確認
- `mysqlrplshow` – レプリケーショントポロジ(親子関係)を図示

サーバ管理

- mysqlserverclone – 既存のMySQLサーバのコピーを作成
- mysqlserverinfo – サーバの稼働状況を表示

MySQL Workbenchのダウンロード

- MySQL Workbench
 - <http://www.mysql.com/downloads/workbench/>
- MySQL Utilitiesのドキュメント:
 - <http://dev.mysql.com/doc/workbench/en/mysql-utilities.html>

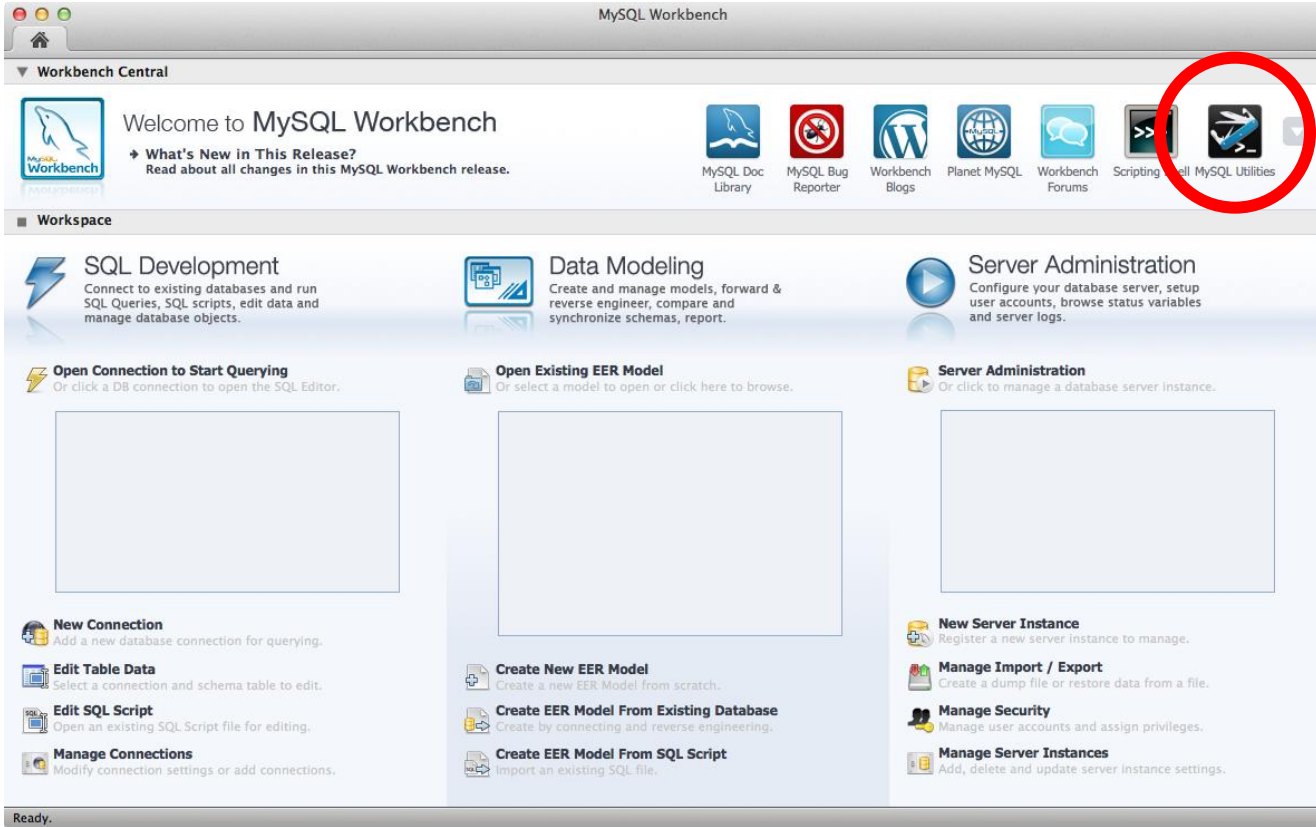
Launchpadからのダウンロード

- <https://launchpad.net/mysql-utilities>
 - `bzr branch lp:mysql-utilities`
- Connector/Pythonが必要
 - <https://launchpad.net/myconnpy>
 - `bzr branch lp:myconnpy`

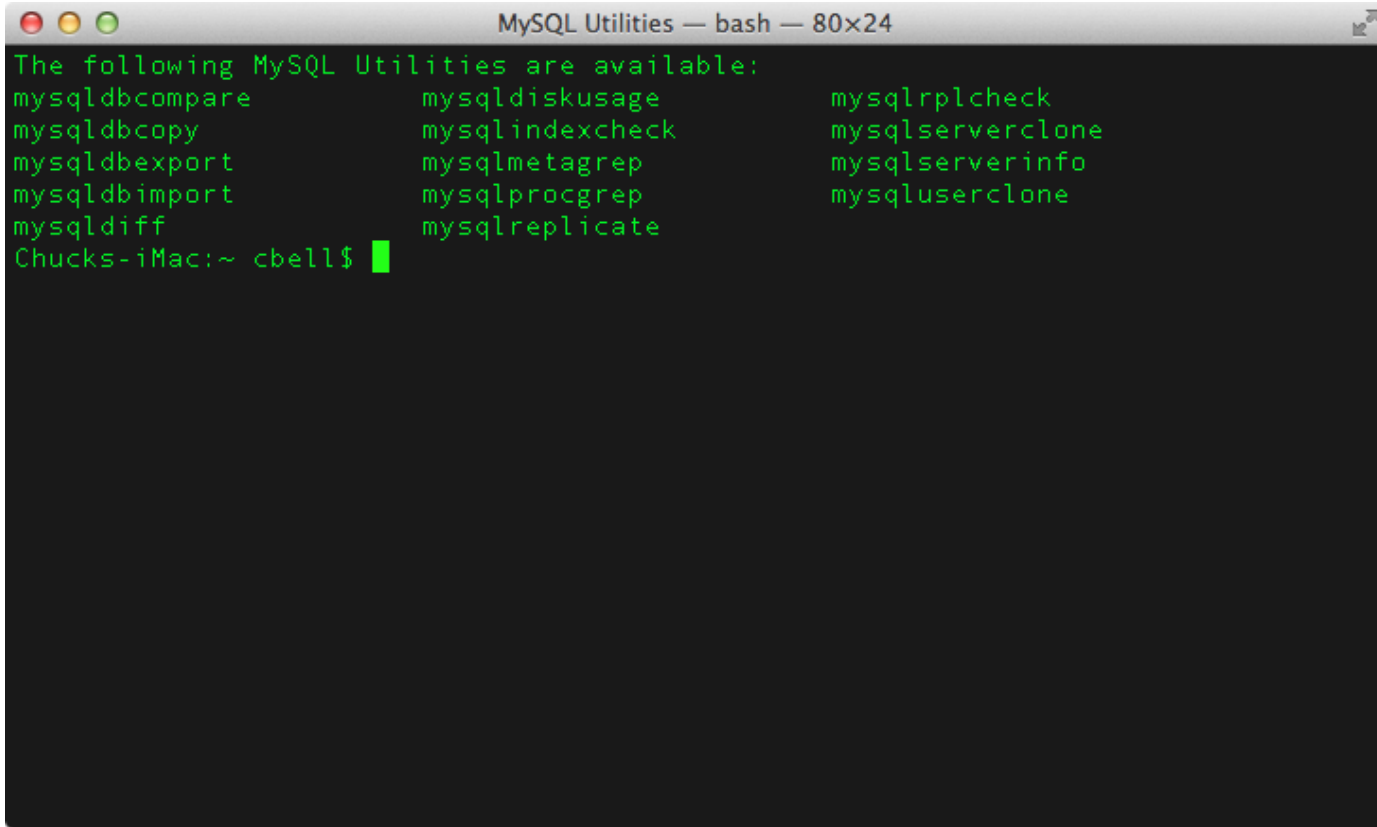
MySQL WorkbenchからMySQL Utilitiesを起動

- MySQL Utilitiesコマンドウィンドウを起動:
 - メニューの Plugins を選択
 - “Start Shell for MySQL Utilities”を選択
- または -
- Workbenchのメイン画面から:
 - メイン画面右側の ▼ をクリック
 - MySQL Utilitiesのアイコンを選択

MySQL WorkbenchからMySQL Utilitiesを起動

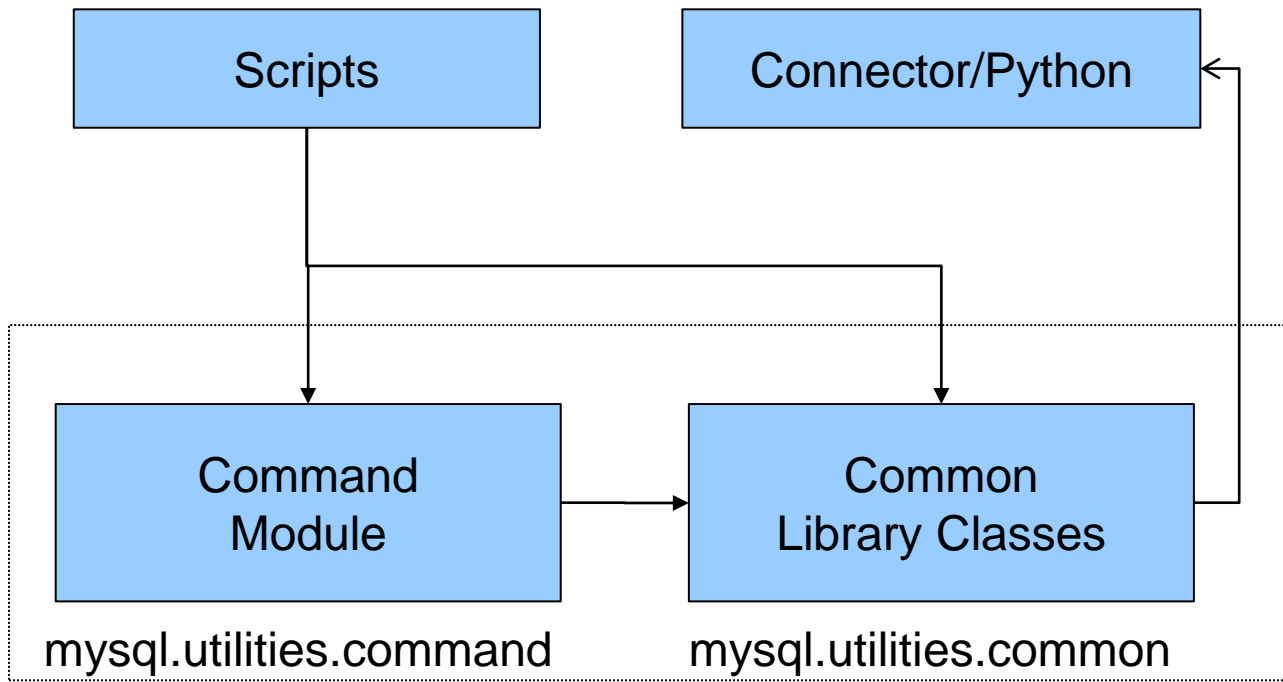


MySQL WorkbenchからMySQL Utilitiesを起動



```
MySQL Utilities — bash — 80x24
The following MySQL Utilities are available:
mysqldbcompare      mysqldiskusage      mysqlrplcheck
mysqldbcopy         mysqlindexcheck     mysqlserverclone
mysqldbexport       mysqlmetagrep       mysqlserverinfo
mysqldbimport       mysqlprogrep        mysqluserclone
mysqldiff           mysqlreplicate
Chucks-iMac:~ cbell$
```

アーキテクチャ



MySQL Utilities Library

mysqlfailoverによる自動フェールオーバー

- GTIDが有効になったMySQL 5.6の自動フェールオーバーが可能
- レプリケーションの稼働状態をチェック
- 1.0.5 での新機能 (MySQL Workbench 5.2.39)

mysqlfailoverによる自動フェールオーバー

```
Terminal — Python — 80x24
MySQL Replication Failover Utility
Failover Mode = auto      Next Interval = Wed Apr  4 11:29:54 2012

Master Information
-----
Binary Log File  Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql-bin.000001  1035

Replication Health Status
+-----+-----+-----+-----+-----+-----+
| host      | port  | role   | state | gtid_mode | health |
+-----+-----+-----+-----+-----+-----+
| localhost | 3310  | MASTER | UP    | ON        | OK    |
| localhost | 3311  | SLAVE  | UP    | ON        | OK    |
| localhost | 3312  | SLAVE  | UP    | ON        | OK    |
| localhost | 3313  | SLAVE  | UP    | ON        | OK    |
| localhost | 3314  | SLAVE  | UP    | ON        | OK    |
| localhost | 3315  | SLAVE  | UP    | ON        | OK    |
| localhost | 3316  | SLAVE  | UP    | ON        | OK    |
| localhost | 3317  | SLAVE  | UP    | ON        | OK    |
| localhost | 3318  | SLAVE  | UP    | ON        | OK    |
+-----+-----+-----+-----+-----+-----+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries Up|Down-scroll
```

mysqlfailover

- フェールオーバーのモード
 - **Auto** – まず候補リストのスレーブ、その後それ以外のスレーブにフェールオーバー
 - **Elect** – 候補リストのスレーブのみにフェールオーバー
 - **Fail** – マスター障害時にエラーとする

mysqlfailover

- 拡張のポイント

- **exec-fail-check** - アプリケーション特有の問題を検知して、フェールオーバーが必要か判断するスクリプトを実行
- **exec-before** - フェールオーバーが起こる前に実行されるスクリプト
- **exec-after** - 新しいマスターにフェールオーバーした直後に実行されるスクリプト
- **exec-post-fail** - フェールオーバーが完了し、全てのスレーブが新しいマスターを参照した後で実行されるスクリプト

実行例 - failover

```
Terminal — Python — 80x24
Failover starting in 'auto' mode...
# Candidate slave localhost:3307 will become the new master.
# Preparing candidate for failover.
# Creating replication user if it does not exist.
# Stopping slaves.
# Performing STOP on all slaves.
# Switching slaves to new master.
# Starting slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Failover complete.
# Discovering slaves for master at localhost:3307

Failover console will restart in 5 seconds.
█
```

mysqlrpladmin によるレプリケーション管理

- レプリケーション環境内のサーバを管理
- レプリケーション接続の稼働状況を確認
- フェールオーバー
- スイッチオーバー (手動でのスレーブ昇格)
- レプリケーション停止などのスレーブの操作

mysqlrpladmin コマンド

- **elect** - (GTID必要) フェールオーバーやスイッチオーバーの際に利用すべきスレーブを選択
- **failover** - (GTID必要) 「最適な」スレーブにフェールオーバー。スレーブのリストから最適なをテストし、選択後は未実行のトランザクションを他のスレーブから収集し、最新の状態とする。その後、新しいマスターに昇格。
- **gtid** - (GTID必要) GTIDの内容を表示する。

mysqlrpladmin コマンド

- **health** - レプリケーション構成の稼働状況を確認
- **reset** - スレーブ上でSTOP SLAVE と RESET SLAVE を実行
- **start** - START SLAVE を実行
- **stop** - 全てのスレーブ上で STOP SLAVE を実行
- **switchover** - --new-master オプションで指定したスレーブを新しいマスターとして昇格させる。GTIDを利用していない環境でも利用可能

実行例 - switchover

```
Terminal — bash — 87x33
# Discovering slaves for master at localhost:3307
# Checking privileges.
# Performing switchover from master at localhost:3307 to slave at localhost:3310.
# Checking candidate slave prerequisites.
# Waiting for slaves to catch up to old master.
# Stopping slaves.
# Performing STOP on all slaves.
# Demoting old master to be a slave to the new master.
# Switching slaves to new master.
# Starting all slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Switchover complete.
# Getting health for master: localhost:3310.
#
# Replication Topology Health:
+-----+-----+-----+-----+-----+
| host      | port  | role   | state | gtid_mode | health |
+-----+-----+-----+-----+-----+
| localhost | 3310  | MASTER | UP    | ON        | OK    |
| localhost | 3308  | SLAVE  | UP    | ON        | OK    |
| localhost | 3309  | SLAVE  | UP    | ON        | OK    |
| localhost | 3311  | SLAVE  | UP    | ON        | OK    |
| localhost | 3312  | SLAVE  | UP    | ON        | OK    |
| localhost | 3313  | SLAVE  | UP    | ON        | OK    |
| localhost | 3314  | SLAVE  | UP    | ON        | OK    |
| localhost | 3315  | SLAVE  | UP    | ON        | OK    |
| localhost | 3316  | SLAVE  | UP    | ON        | OK    |
| localhost | 3317  | SLAVE  | UP    | ON        | OK    |
| localhost | 3307  | SLAVE  | UP    | ON        | OK    |
+-----+-----+-----+-----+-----+
# ...done.
Chucks-iMac:mysql-wl-6143 cbell$
```

レプリケーションの構築と状況確認

- 既存のレプリケーションが存在する状態で、新たにスレーブを追加する。スレーブとなるサーバにはMySQLはインストール済み。
 1. `mysqldbexport` でマスターのデータをエクスポート
 2. `mysqldbimport` で新しいスレーブにデータをインポート
 3. `mysqluserclone` で新しいスレーブにユーザアカウントをコピー
 4. `mysqlreplicate` で新しいスレーブにレプリケーションを設定
 5. `mysqlrpladmin` でレプリケーション構成を確認
- フェールオーバーは `mysqlfailover` を利用

Try NOW!!

MySQL 5.6 GA

& MySQL Utilities



MySQL™

Hardware and Software

ORACLE®

Engineered to Work Together

ORACLE®