

# [13주차] 17.1 ~ 17.3-

☰ 태그	Done
📅 날짜	@2024년 4월 23일 → 2024년 4월 30일
☰ 제목	InnoDB 클러스터

## 📌 17장 InnoDB 클러스터

- ✅ 17.1 InnoDB 클러스터 아키텍처
- ✅ 17.2 그룹 복제(Group Replication)
  - [17.2.1 그룹 복제 아키텍처](#)
  - [17.2.2 그룹 복제 모드](#)
  - [17.2.3 그룹 멤버 관리\(Group Membership\)](#)
  - [17.2.4 그룹 복제에서의 트랜잭션 처리](#)
  - [17.2.5 그룹 복제의 자동 장애 감지 및 대응](#)
  - [17.2.6 그룹 복제의 분산 복구](#)
  - [17.2.7~8 그룹 복제 요구사항 및 그룹 복제 제약 사항](#)
- ✅ 17.3 MySQL 헬

## 📌 17장 InnoDB 클러스터

복제 구성시 소스 서버에서의 장애 발생시 자동으로 레플리카 서버가 소스 서버로 승격되지 않습니다. (자체적인 Fail over 처리가 없음)

이런 문제점을 해결하기 위해 솔루션을 사용해왔는데 5.7.17버전부터 빌트인 형태의 HA 솔루션인 InnoDB 클러스터가 도입되어 사용자가 좀 더 쉽고 편하게고가용성을 실현할 수 있게 되었습니다.

### ✅ 17.1 InnoDB 클러스터 아키텍처

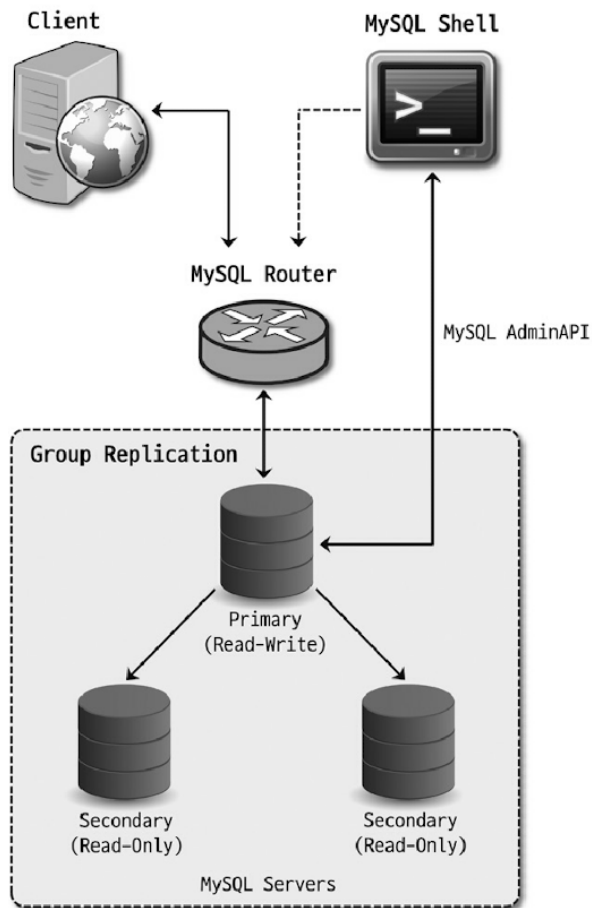


그림 17.1 InnoDB 클러스터 구조

## • 구조 설명

### ○ InnoDB 클러스터

고가용성 실현을 위해 만들어진 여러 구성 요소들의 집합체

### ○ 그룹 복제 (Group Replication)

기본적인 복제 + 복제에 참여하는 mysql 서버들에 대한 자동화된 멤버십 관리(새로운 멤버의 추가 및 제거) 역할을 담당

### ○ MySQL Router

애플리케이션 서버와 mysql 서버 사이에서 동작하는 **미들웨어 프로그램**, 애플리케이션이 실행한 쿼리를 적절한 mysql 서버로 전달하는 **Proxy** 역할을 합니다.

### ○ MySQL Shell

기존 mysql 클라이언트보다 확장된 기능을 가진 새로운 클라이언트 프로그램, SQL문 실행 + JS, Python 기반 스크립트 작성, mysql 서버에 대한 클러스터 구

성 등의 어드민 작업을 할 수 있게 하는 API를 제공

- 특징

- 소스 서버, 레플리카 서버 → 프라이머리, 세컨더리로 불림
  - 복제 그룹 내에서는 InnoDB 스토리지 엔진만 사용될 수 있음
  - 최소 3대 이상으로 구성해야 함 → 서버 한 대에 장애가 발생해도 복제 그룹이 정상적으로 동작하기 때문
  - 클라이언트는 mysql 라우터에 연결해 쿼리를 실행함 (직접 접근 하지 않음)
    - 클라이언트는 어떤 서버로 구성돼 있는지 알 필요가 없고, 커넥션에는 라우터 서버만 설정하면 됩니다.
  - mysql 쉘에서 InnoDB 클러스터와 관련된 작업을 하려면 클러스터 내 mysql 서버에 직접 연결해 작업해야 함
    - 단순히 쿼리를 실행하는 경우 mysql 라우터로 연결해 처리함
  - 클러스터에서 장애가 발생하면 그룹 복제가 자동으로 감지하고 해당 서버를 복제 그룹에서 제외 시킵니다.
    - 라우터는 이를 인지하고 자신의 메타데이터를 갱신 및 정상적인 mysql 서버로만 쿼리를 전달하도록 합니다.
  - InnoDB 클러스터는 수동으로 처리하던 장애 복구 처리를 자동으로 처리해줍니다.
  - InnoDB 클러스터는 mysql에서 빌트인 HA 솔루션을 제공해 하나의 패키지로 개발 및 테스트되는것이 큰 강점임
- 

## 17.2 그룹 복제(Group Replication)

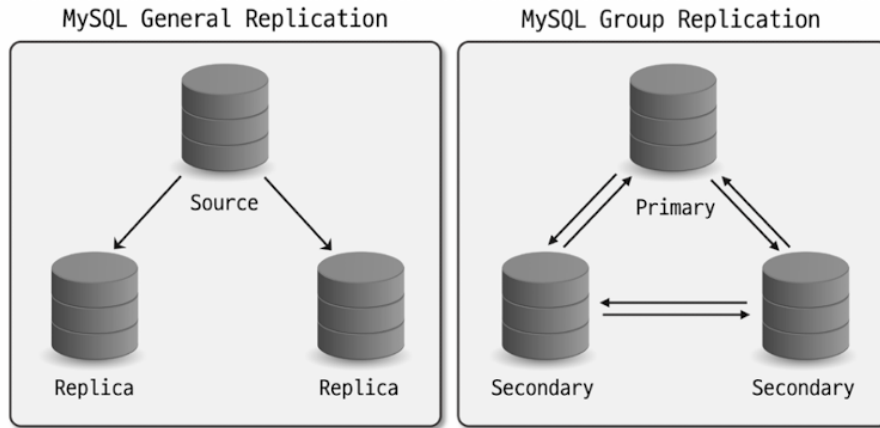


그림 17.2 MySQL의 일반적인 복제와 그룹 복제 동작 방식 비교

- 내부적으로 Row 포맷인 바이너리 로그 및 릴레이 로그, GTID를 사용합니다.
- 기존 복제와 그룹 복제는 유사하지만 복제 구성 형태 및 트랜잭션 처리 방식 측면에선 완전히 다릅니다.
  - 기존 복제처럼 단방향인 아니라 서로 통신하며 양방향으로도 복제 처리가 가능함 덕분에 하나의 복제 그룹 내에서 쓰기를 처리하는 서버가 여러대 존재할 수 있음
  - 복제 처리 방식은 반동기와 비슷하지만 다른 부분이 존재합니다.

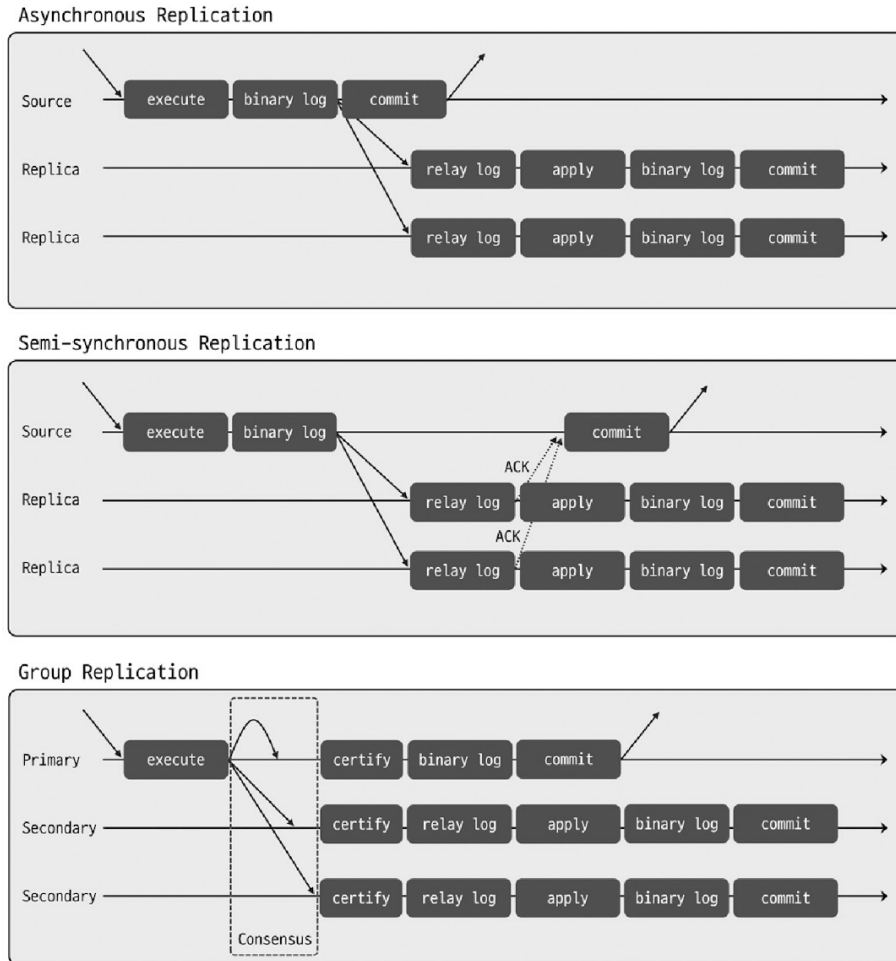


그림 17.3 데이터 동기화 방식별 복제 처리 방식 비교

- 한 서버에서 트랜잭션이 커밋될 준비가 되면 해당 트랜잭션 정보를 다른 멤버들에 전송합니다. 이때 과반수 이상의 멤버로부터 응답을 전달받으면 그때 해당 트랜잭션을 인증(Certify)하고 최종적으로 커밋 처리를 완료합니다.
  - 과반수가 안된다면 해당 트랜잭션은 그룹에 적용되지 않는데 트랜잭션이 최 발생한 서버에서 트랜잭션 커밋 처리시 멤버들의 트랜잭션 적용 여부까지는 확인하지 않는다는 점에서 기존 반동기와 차이점이 존재합니다.
  - 기존에는 일방적 적용이었다면 그룹 복제에선 다른 멤버들에게 동의를 구하는 합의 과정 이후 적용이 됩니다. (읽기의 경우 노상관)
- 그룹 복제에 참여하는 mysql 서버들을 **그룹 멤버** 라 지칭합니다.
  - **그룹 복제에서 제공하는 대표적 기능**
    - 그룹 멤버 관리
    - 그룹 단위의 정렬된 트랜잭션 적용 및 트랜잭션 충돌 감지

- Auto Fail Over
- 자동 분산 복구

## 17.2.1 그룹 복제 아키텍처

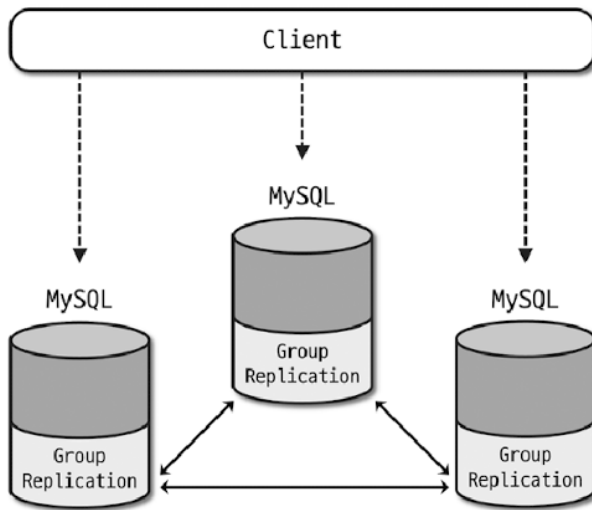


그림 17.4 그룹 복제의 기본 구조

- 그룹 복제는 별도의 플러그인으로 구현돼 있기에 mysql 서버에 그룹 복제 플러그인이 설치되어 있어야 합니다.
- 플러그인은 복제 설정이 완료되면 `group_replication_applier` 라는 복제 채널을 생성해 해당 채널을 통해 그룹에서 실행된 모든 트랜잭션을 전달받아 적용합니다.
- 플러그인은 새로운 서버가 추가되어 그룹 복제 분산 복구 작업이 필요하다면 `group_replication_recovery` 복제 채널을 생성해 분산 복구 작업을 진행합니다.
- 그룹 복제는 구성하는 mysql 서버 수가 중요한데 과반수에 해당하는 서버가 정상적으로 동작해야 하나의 서버에 장애가 발생해도 그룹 복제가 문제없이 처리됩니다.  
(적어도 3대는 필요함)
  - 이를 공식으로 표현하면:  $n = 2f + 1$  (n: 전체 서버 수, f: 허용하고자 하는 장애 서버 수)

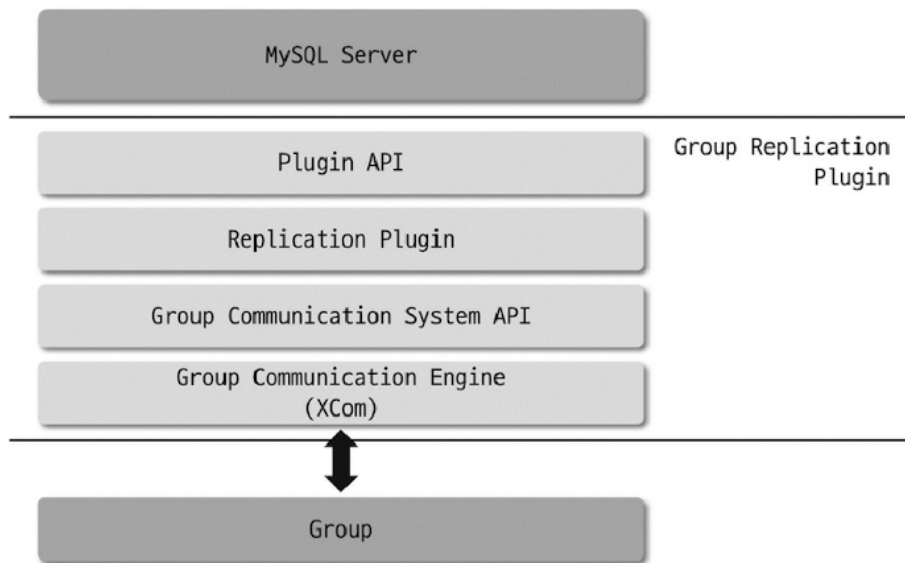


그림 17.5 그룹 복제 플러그인의 내부 구조

- **Plugin API**
  - mysql 서버와 상호작용하기 위해 구현된 인터페이스
  - API를 통해 **mysql 서버 ↔ 그룹 복제 플러그인**에 양방향으로 요청을 전달하고 받습니다.
  - 이벤트를 그룹 복제 플러그인에 전달하고 그룹 복제 플러그인에선 처리 중인 트랜잭션에 대한 커밋, 중단, 릴레이 로그 기록 요청 등을 서버에 전달함
- **Replication Plugin**
  - 그룹 복제의 기능들이 실질적으로 구현돼 있음
  - 계층 내부는 여러 모듈로 나뉘져 있으며 API를 통해 들어온 요청을 적절한 모듈로 전달함
  - **로컬 및 그룹 복제의 다른 mysql 서버**에서 실행된 원격 트랜잭션들이 처리되며 트랜잭션들에 대한 충돌 감지 및 그룹 내 전파 등이 수행됩니다.  
(그룹 복제의 분산 복구 작업도 처리됨)
- **Group Communication System API, Group Communication Engine(XCom)**
  - 그룹 통신 시스템 API를 통해 상위 플러그인 계층 ↔ 그룹 통신 엔진간 상호작용을 함
  - 그룹 통신 엔진은 그룹 복제에 참여중인 다른 mysql 서버들과의 통신 처리를 담당하는 그룹 복제의 핵심 구성 요소
    - 트랜잭션이 그룹 복제 멤버들에 동일한 순서로 전달될 수 있도록 보장

- 그룹 복제 토폴로지의 변경, 멤버의 장애 등을 감지
- 트랜잭션 적용 위한 멤버간 합의 처리 담당
  - 합의 처리를 위해 사용하는 대표적 알고리즘으로 Paxos와 Raft가 있습니다.  
(Paxos는 데이터 변경이 서버 여러대에서, Raft는 데이터 변경이 서버 한대에서 발생하는 경우 사용)

## 17.2.2 그룹 복제 모드

쓰기 처리를 한 곳에서만 하는 싱글 프라이머리 모드, 여러곳에서 처리할 수 있는 멀티 프라이머리 두 가지 동작 모드가 존재합니다.

(모드를 변경하려면 그룹 복제를 중단해야 하지만, 8.0.13이상 버전에선 UDF를 통해 중단 없이 그룹 모드를 변경할 수 있습니다.)

### 1. 싱글 프라이머리 모드

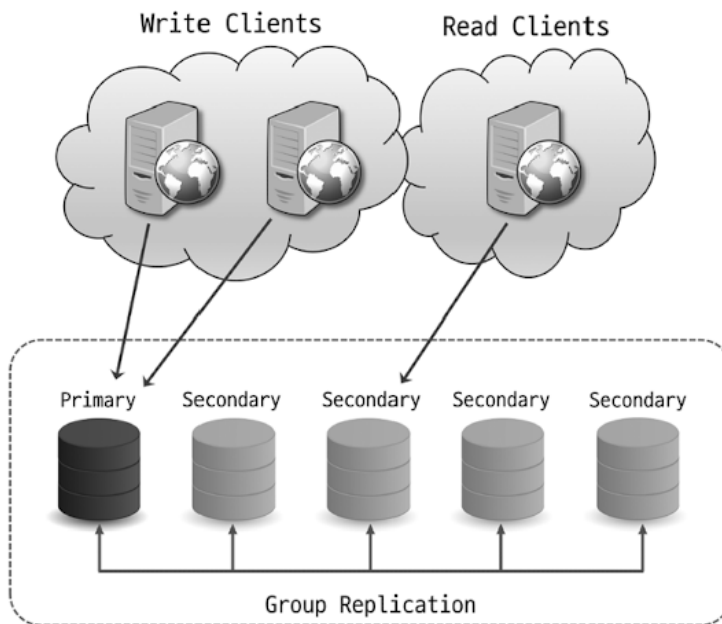


그림 17.6 싱글 프라이머리 모드로 설정된 그룹 복제

- 그룹 내에서 쓰기 처리를 할 수 있는 프라이머리 서버가 한 대만 존재함
- 처음 구축하는 경우 그룹 복제 구축을 진행한 mysql 서버가 프라이머리로 지정됨



- 다른 서버들이 그룹에 참여하는 시점에 자동으로 `super_read_only`가 ON되어 읽기 전용 모드로 동작합니다.
- 다음과 같은 상황에서 그룹내 프라이머리 서버가 변경됩니다.
  1. 자발적, 예기치 않게 기존 프라이머리 서버가 그룹 탈퇴할때
  2. `group_replication_set_as_primary()` UDF로 특정 멤버를 새로운 프라이머리로 지정한 경우 (8.0.13이상)
- UDF로 지정하는 경우를 제외하고 예기치 못한 상황에선 다음과 같은 기준으로 선정합니다.
  1. **버전 체크** : 가장 낮은 버전을 우선시함, 모두 8.0.17이상이면 패치 버전을 기준으로 아니라면 메이저 버전을 기준으로 정렬함
  2. **각 멤버의 가중치 값** : 동일한 버전이라면 더 높은 가중치를 선정합니다. 이는 별도로 시스템 변수에 지정할 수 있습니다.  
(다만 5.7.20버전 부터 가중치 변수가 도입됐기에 해당 버전 미만이라면 가중치 기준은 무시됨)
  3. **UUID 값의 사전식 순서** : 가중치까지 동일하면 UUID값을 사전순 정렬해 가장 낮은 값을 가진 멤버를 선택합니다.

## 2. 멀티 프라이머리 모드

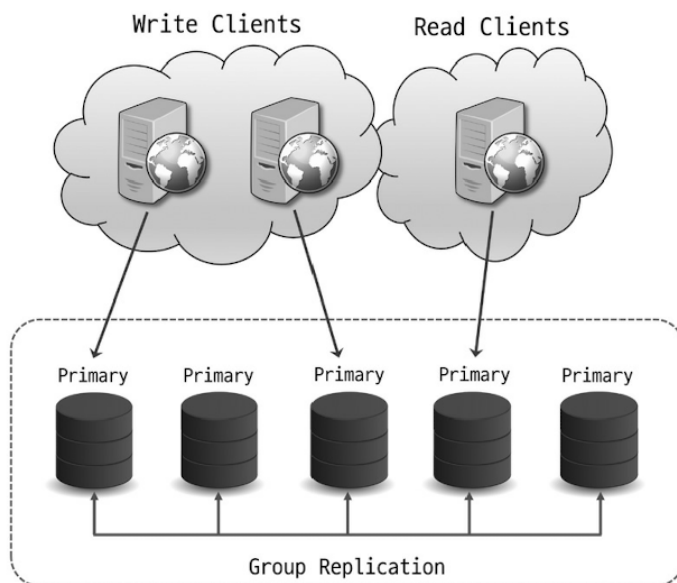


그림 17.7 멀티 프라이머리 모드로 설정된 그룹 복제

- 그룹 멤버들이 전부 프라이머리로 동작하는 형태
- 각 멤버에서 발생한 쓰기는 다른 멤버에게 전파되며, 그룹 멤버간 mysql 버전 호환이 정말 중요합니다. 따라서 그룹 복제에 참여할때 버전 호환성을 확인하는 기능이 구현돼 있습니다.
- 그룹 복제 새로 참여시 호환성 검사를 위한 기준
  1. 새로운 멤버가 그룹에 존재하는 가장 낮은 mysql 버전보다 낮은 버전이라면 참여 불가
  2. 기존 멤버의 가장 낮은 mysql버전과 동일한 버전이면 그룹 참여 가능
  3. 가장 낮은 mysql 버전보다 높은 mysql 버전을 사용중이라면 그룹 참여는 가능하지만 읽기 전용 모드를 유지함
- 싱글 → 멀티 변환을 하면
  - 그룹 내 멤버들이 mysql 버전을 바탕으로 자동으로 적절한 모드로 설정함

### 17.2.3 그룹 멤버 관리(Group Membership)

어떤 서버들이 그룹에 참여하는지, 멤버들에 대한 목록 및 상태 정보를 `performance_schema.replication_group_members` 에서 관리하고 있습니다.

```

CHANNEL_NAME: group_replication_applier
  MEMBER_ID: a546685c-639a-11eb-ba24-77f8776aaf30
  MEMBER_HOST: ic-node3
  MEMBER_PORT: 3306
  MEMBER_STATE: ONLINE
  MEMBER_ROLE: SECONDARY
MEMBER_VERSION: 8.0.22
  
```

- MEMBER\_STATE는 `ONLINE`, `RECOVERING`, `OFFLINE`, `ERROR`, `UNREACHABLE` 와 같은 상태가 존재합니다.
- 그룹 복제과 관리하는 멤버 목록 및 상태 정보를 View라고 합니다. 언제든지 가입 및 탈퇴를 반복할 수 있으므로 `뷰는 특정 시점의 그룹 멤버 목록` 입니다.

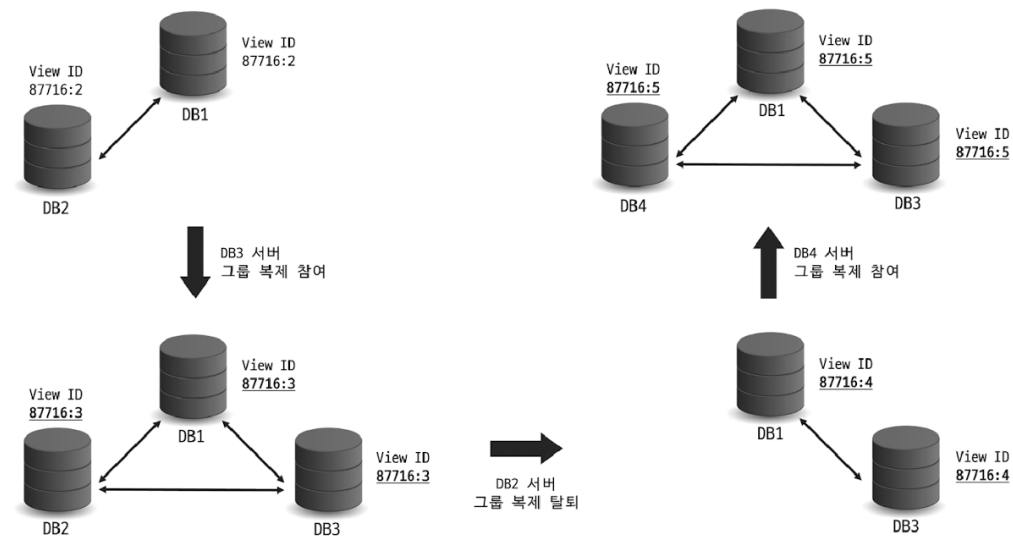


그림 17.8 그룹 복제의 뷰 ID 변경 과정

- View ID는 고유 식별자이며 `[PrefixValue] : [Sequence Value(그룹 멤버 변경 있을때마다 단조 증가)]` 와 같이 구성됩니다.
- 뷰의 변경은 View\_change라는 이벤트로 바이너리 로그에도 변경 내역이 기록됩니다.

## 17.2.4 그룹 복제에서의 트랜잭션 처리

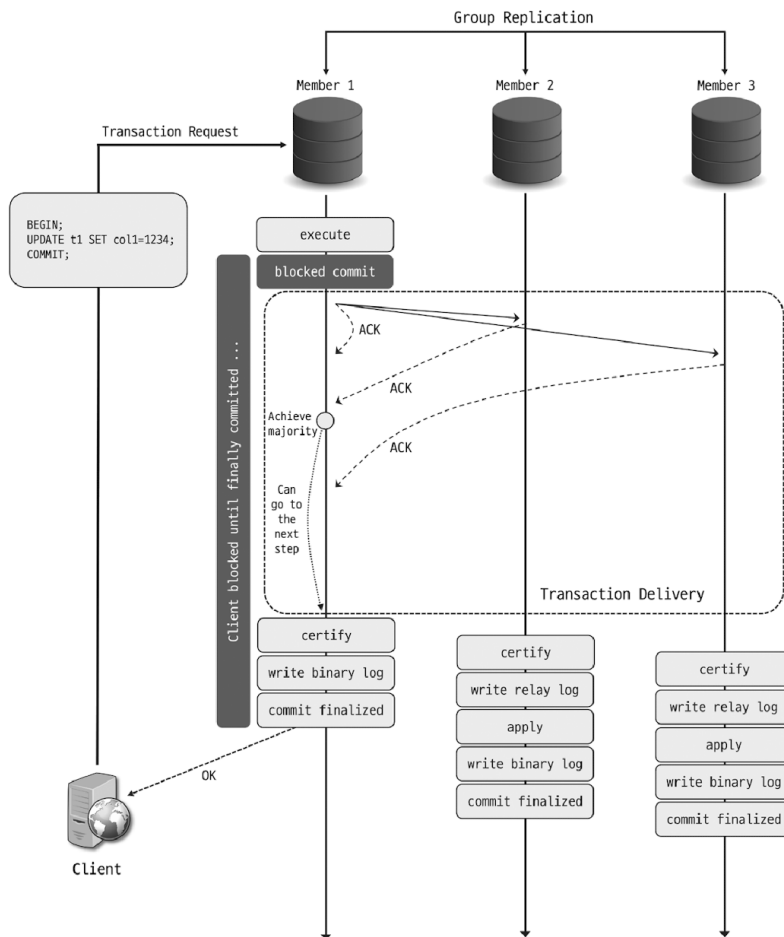


그림 17.9 그룹 복제에서의 트랜잭션 처리 과정

그룹 복제에서의 트랜잭션은 **합의(Consensus)**, **인증(Certification)** 단계를 거치고 그룹의 각 서버에 적용됩니다.

### 1. 합의(Consensus)

- 그룹 멤버들에게 트랜잭션 적용을 제안하고 승낙을 받는 과정 (그룹내 일관된 트랜잭션 적용 위해)
- 그룹 멤버간 통신 결과를 바탕으로 처리됨
- 한 멤버에서 클라이언트의 트랜잭션이 실행하고 커밋 요청을 보내면 XCom을 통해 트랜잭션에서 변경한 트랜잭션 데이터(WriteSet, gtid\_executed 스냅샷, 이벤트 로그 데이터 등)를 정상 동작중인 다른 멤버에게 전파합니다.
- 과반수 이상 멤버들에게 ACK를 받으면 해당 멤버는 그 다음 프로세스를 진행합니다.

### 2. 인증(Certification)

- 합의 이후 글로벌하게 정렬되고 각 멤버들에서 모두 동일한 순서로 인증 단계를 거칩니다.

- 전달받은 트랜잭션 WriteSet과 로컬에서 내부 관리하는 WriteSet 히스토리 데이터를 바탕으로 트랜잭션 충돌 여부를 파악합니다.  
(충돌은 여러곳에서 쓰기가 가능한 멀티 프라이머리 모드에서만 발생함)
  - 충돌이 감지되면 롤백됩니다.

인증 단계까지 거치면 바이너리 로그에 트랜잭션을 기록하고 최종적으로 커밋을 완료하고, 클라이언트는 이 시점에 커밋 요청에 대한 응답을 받습니다.

다른 멤버들도 인증 단계 수행 이후 전달받은 트랜잭션 데이터 기반으로 릴레이 로그 이벤트를 작성합니다.

그룹 복제의 Applier Thread에선 릴레이 로그에 기록된 트랜잭션을 실행하고 바이너리 로그에도 기록해 최종적으로 서버에 해당 트랜잭션을 적용합니다.

## 1. 트랜잭션의 일관성 수준

다른 멤버에 바로 적용되지 않는 경우도 있기 때문에 데이터를 읽었을때 최신 데이터가 아닐 수 있습니다. 이런 맹점을 보완하기 위해 **8.0.14 버전부터 그룹 복제에서의 트랜잭션 일관성 수준을 설정할 수 있습니다.**

다만 설정된 일관된 수준은 읽기 전용 트랜잭션, 읽기-쓰기 트랜잭션에 서로 다른 영향을 미치고, 다른 멤버에서 실행중인 트랜잭션에도 영향을 줄 수 있음을 알아둬야 합니다.

### 1-1. EVENTUAL 일관성 수준

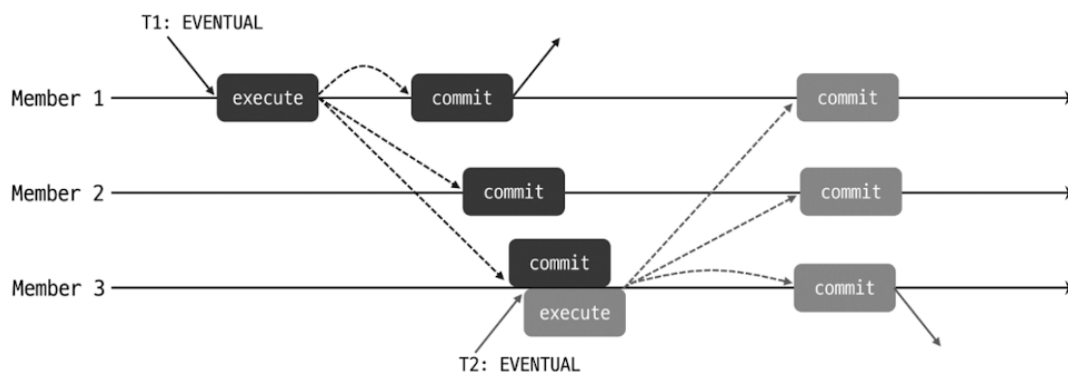


그림 17.10 EVENTUAL 일관성 수준에서의 트랜잭션 처리

- 기본 설정값

- 읽기 전용 및 읽기-쓰기 전용 트랜잭션이 별도의 제약없이 바로 실행함 따라서 변경 이전의 데이터가 읽힐 수 있음, 또한 읽기-쓰기 트랜잭션에서 충돌이 발생해 롤백이 발생할 수 있음

## 1-2. BEFORE\_ON\_PRIMARY\_FAILOVER 일관성 수준

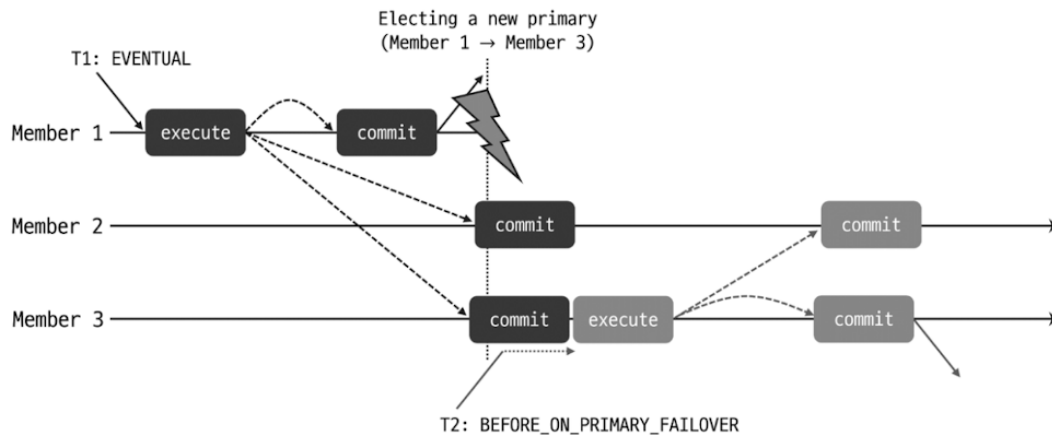


그림 17.11 BEFORE\_ON\_PRIMARY\_FAILOVER 일관성 수준에서의 트랜잭션 처리

BEFORE\_ON\_PRIMARY\_FAILOVER 일관성 수준에서는 다음과 같은 부분들이 보장된다.

- 신규 프라이머리로 유입된 읽기 전용 및 읽기-쓰기 트랜잭션들은 오래된 데이터가 아닌 최신 데이터를 바탕으로 동작하게 된다.
- 신규 프라이머리로 유입된 읽기-쓰기 트랜잭션은 적용 대기 중인 이전 프라이머리의 트랜잭션과의 충돌로 롤백될 수도 있는데, BEFORE\_ON\_PRIMARY\_FAILOVER 일관성 수준을 사용하면 이 같은 롤백은 발생하지 않게 된다.

- 싱글 프라이머리 모드에서 프라이머리 페일오버 발생시 새로운 프라이머리가 선출됐을 때만 트랜잭션에 영향을 줌
- 이전 프라이머리의 트랜잭션 적용중에 새로운 프라이머리로 유입된 읽기-쓰기 트랜잭션은 적용중이던 트랜잭션이 완료될때까지 대기합니다.
- 이런 지연 때문에 애플리케이션 단에서는 지연을 대비하는 코드가 구현돼 있는게 좋습니다.
- `wait_timeout` 시스템 변수에 설정된 값을 통해 대기 시간을 설정함
- 서버 모니터링 위한 일부 쿼리는 즉시 실행 가능함
- 프라이머리 페일오버가 없다면 EVENTUAL 일관성 수준으로 설정된 트랜잭션처럼 처리됨

### 1-3 BEFORE 일관성 수준

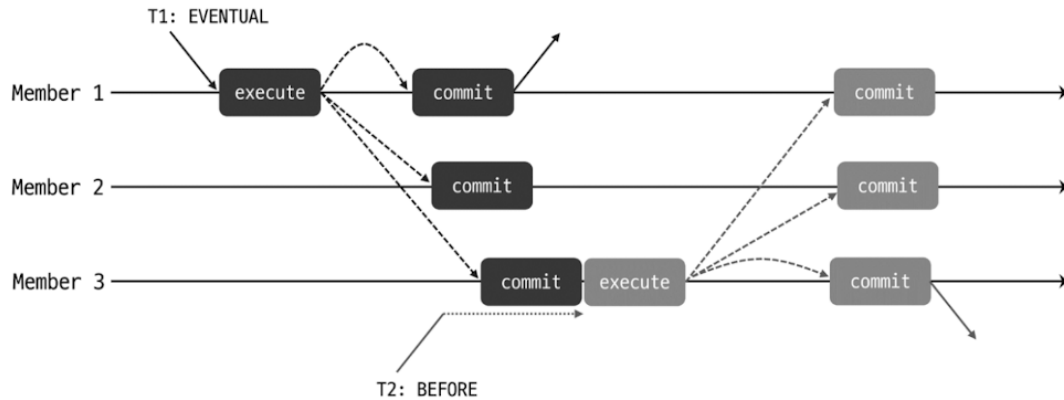


그림 17.12 BEFORE 일관성 수준에서의 트랜잭션 처리

- 읽기, 읽기-쓰기 트랜잭션은 모든 선행 트랜잭션이 완료될때까지 대기 후 처리됨
- 따라서 모든 트랜잭션은 항상 최신 데이터를 읽습니다.
- DB에서 읽기 요청은 적고 쓰기 요청이 많은 경우 사용하는게 좋습니다.
- BEFORE\_ON\_PRIMARY\_FAILOVER 수준이 제공하는 일관성 보장을 포함함

### 1-4 AFTER 일관성 수준

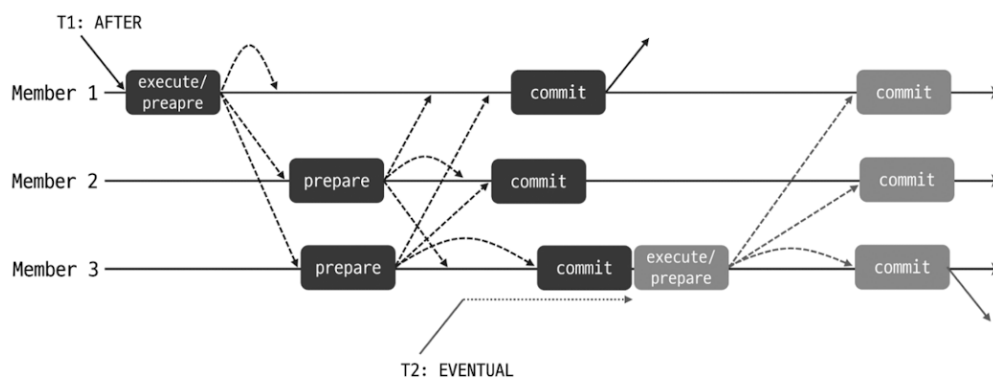


그림 17.13 AFTER 일관성 수준에서의 트랜잭션 처리

- 트랜잭션이 적용되면 해당 시점에 그룹 멤버들이 모두 동기화된 데이터를 갖게 합니다.

- 읽기-쓰기 트랜잭션은 다른 모든 멤버들에서 해당 트랜잭션이 커밋될 준비가 될때까지 대기하고 최종 처리됩니다. (읽기는 즉시 처리됨)
  - 따라서 다른 멤버에서 동시점에 실행되는 트랜잭션에 영향을 미칩니다.
  - 따라서 항상 본래 처리시간보다 더 많은 시간을 소요합니다.
- 읽기 요청이 많고 분산된 최신 읽기를 해야 하는 경우 사용하는게 좋음
- BEFORE\_ON\_PRIMARY\_FAILOVER 수준이 제공하는 일관성 보장을 포함함

## 1-5. BEFORE\_AND\_AFTER 일관성 수준

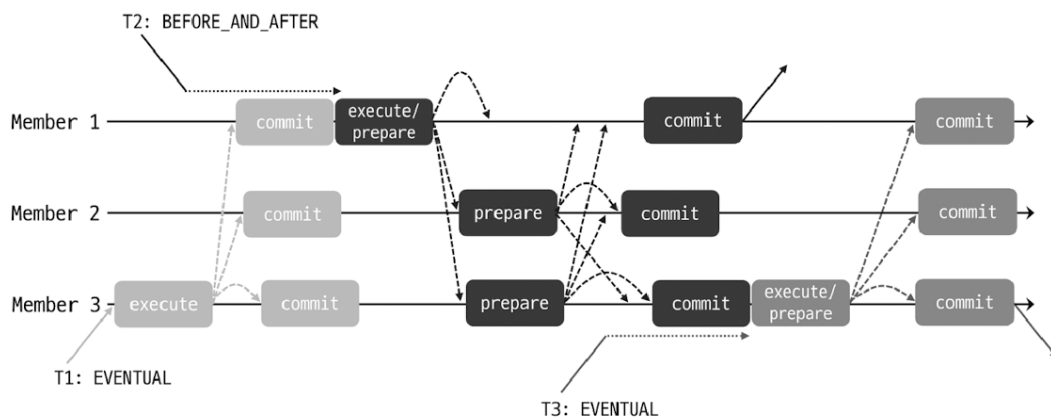


그림 17.14 BEFORE\_AND\_AFTER 일관성 수준에서의 트랜잭션 처리

- BEFORE + AFTER 수준이 결합된 형태
- 읽기-쓰기, 읽기 트랜잭션 모두 모든 선행 트랜잭션이 적용될때까지 기다리고 실행하고 모든 멤버가 준비완료되어야 최종 커밋이 됨
- BEFORE\_ON\_PRIMARY\_FAILOVER 수준이 제공하는 일관성 보장을 포함함

## 2. 흐름제어(Flow Control)

- 그룹 멤버간 하드웨어, 네트워크 대역폭의 차이로 인해 부하의 불균형을 방지하기 위해 그룹 멤버들의 쓰기 처리량을 조절하는 메커니즘을 말합니다.
- 멤버 간 트랜잭션 갭을 적게 유지해 멤버들의 데이터가 최대한 동기화된 상태로 유지할 수 있게 하는게 핵심입니다.
- 현재는 **QUOTA 모드** 밖에 존재하지 않는데 다음과 같이 동작합니다.



1. 멤버들의 쓰기 처리량 및 대기중인 트랜잭션에 대한 통계를 수집해 멤버의 처리량을 조절할 필요성을 파악
    - 필요성은 인증 큐, 적용 큐 크기를 바탕으로 판단합니다.
  2. 조절해야 한다면 통계 데이터를 바탕으로 멤버에게 할당할 쓰기 처리량을 계산하고 최대 쓰기 처리량을 넘어 쓰기를 처리하지 않도록 멤버의 쓰기 처리를 제한합니다.
- 흐름제어는 각 멤버에서 개별 수행되고 아래 통계 정보를 수집해 다른 멤버들에게도 공유합니다.
    - 인증 큐 크기, 적용 큐 크기, 인증된 총 트랜잭션 수, 적용된 총 트랜잭션 수, 로컬 트랜잭션 수
  - 흐름제어는 매 초마다 동작하는데 통계 정보는 시스템 변수를 통해 수집 시간을 조절할 수 있습니다.
  - 그 밖에도 각 단계별 흐름제어가 처리량 조절을 시작하는 값들이나 최소 쓰기 처리량과 같은 값들을 조절할 수 있습니다.
- 

### 17.2.5 그룹 복제의 자동 장애 감지 및 대응

- 장애 감지 메커니즘은 문제 상태에 있는 멤버를 식별하고 해당 멤버를 그룹 복제에서 제외시킴으로써 그룹이 정상적으로 동작 중인 멤버로만 구성될 수 있게하고 클라이언트의 요청을 문제없이 처리하는걸 말합니다.
- 멤버로부터 일정 시간내(시스템 변수로 조절 가능)로 메시지를 받지 못하면 문제가 생긴 것으로 간주하고, 멤버 과반수가 동의하면 해당 멤버를 추방시킵니다.
- **그룹 멤버의 추가, 삭제가 된다면 그룹 뷰 ID가 변경됩니다.**
- 추방된 멤버는 재가입을 시도할 수 있고, 이 또한 시스템 변수로 횟수를 지정할 수 있습니다.
- 네트워크 단절과 같은 이슈로 인해 그룹 멤버가 분리되면 추방된 멤버는 다시 통신이 되지 않기에 자신의 추방 사실을 모릅니다. 따라서 과반수의 그룹 멤버들과 통신이 단절됐을때 일정 시간 대기하고 스스로 그룹을 탈퇴하도록 설정할 수 있습니다.(시스템 변수를 통해)
  - 이때 실행되던 트랜잭션은 타 멤버의 동의를 구할 수 없으므로 보류상태로 남습니다.
- 통신 단절, 자의, 타의로 그룹에서 탈퇴하고 재가입을 실패하거나 시도하지 않게 설정됐다면 종료 액션을 실행합니다. (시스템 변수로 설정할 수 있음)

- READ\_ONLY

`super_read_only` 시스템 변수를 ON으로 설정해서 MySQL 서버를 슈퍼 읽기 전용 모드로 전환시킨다. 슈퍼 읽기 전용 모드에서는 사용자가 `CONNECTION_ADMIN` 권한(또는 `SUPER` 권한)을 가지고 있더라도 서버에서 데이터 변경 작업을 수행할 수 없다. MySQL 8.0.16 버전부터 `group_replication_exit_state_action` 시스템 변수의 기본 값이 이 값으로 설정됐다. 또한 이 값은 `group_replication_exit_state_action` 시스템 변수가 도입되기 전에 MySQL 8.0에서 수행했던 동작 방식이기도 하다.

- OFFLINE\_MODE

`offline_mode` 시스템 변수를 ON으로 설정해서 MySQL 서버를 오프라인 모드로 전환시키고 `super_read_only` 시스템 변수도 ON으로 설정한다. 오프라인 모드에서는 기존에 이미 연결돼 있는 세션의 경우 다음 요청에서 연결이 끊어지고 `CONNECTION_ADMIN` 권한(또는 `SUPER` 권한)을 가진 사용자를 제외하고 더 이상 연결이 허용되지 않는다. `OFFLINE_MODE` 값은 MySQL 8.0.18 버전부터 설정할 수 있다.

- ABORT\_SERVER

MySQL 서버를 종료시킨다. 이 값은 `group_replication_exit_state_action` 시스템 변수가 도입된 MySQL 8.0.12 버전부터 8.0.15 버전까지 해당 변수의 기본값이었다.

---

- 종료 액션은 다음과 같이 구체적인 경우에 동작합니다.

- 그룹 복제의 어플라이어 스레드(Replicator thread)에 에러가 발생한 경우
- 멤버가 그룹 복제의 분산 복구 프로세스를 정상적으로 완료할 수 없는 경우  
(분산 복구에 대해서는 17.2.6절 ‘그룹 복제의 분산 복구’를 참고하자.)
- `group_replication_switch_to_single_primary_mode()` 같은 그룹 복제 UDF를 사용해 그룹 전체에 대한 설정을 변경하는 중에 에러가 발생한 경우
- 싱글 프라이머리 모드의 그룹에서 새 프라이머리 선출 과정 중 에러가 발생한 경우
- 과반수 이상의 다른 그룹 멤버들과 통신이 단절되고, `group_replication_unreachable_majority_timeout` 시스템 변수에 설정된 대기 시간이 초과됐으나 그룹에 재가입 시도를 하지 않도록 설정된 경우
- 멤버에 문제가 발생해서 `group_replication_member_expel_timeout` 시스템 변수에 설정된 대기 시간이 초과된 후 그룹에서 추방되고 나서, 다시 그룹의 다른 멤버들과 통신이 재개되어 자신이 추방됐음을 확인했으나 그룹에 재가입 시도를 하지 않도록 설정된 경우
- 멤버가 자의 혹은 타의로 그룹에서 탈퇴한 후 `group_replication_autorejoin_tries` 시스템 변수에 지정된 횟수 동안 그룹에 재가입을 성공하지 못한 경우

---

## 17.2.6 그룹 복제의 분산 복구

- 특정 멤버가 탈퇴 후 재가입, 잠시 떠나있는 상황에서 그룹에 적용된 트랜잭션, 즉 누락된 트랜잭션을 다른 그룹 멤버(기증자:Donor)로부터 가져와 적용하는 자동 복구 프로세스를 **분산 복구** 라고 합니다.

- 이때 기증자는 온라인 상태로 존재하는 모든 멤버가 될 수 있습니다.

## 1. 분산 복구 방식

- 분산 복구 방식으로는 2가지 방식이 존재하고, 자동으로 적합한 형태의 복구 방식을 선택합니다.
- 특정 방식의 사용 여부는 threshold값을 시스템 변수를 통해 임곗값을 지정할 수 있습니다.
- 이전에 그룹에 가입했던 멤버고, 릴레이 로그에 아직 적용되지 않은 트랜잭션이 있다면 이를 먼저 적용하는 것으로 복구 작업을 시작합니다.

### 1-1. 바이너리 로그 복제 방식

- 비동기 복제를 기반으로 구현 기증자로 선택된 멤버와 `group_replication_recovery` 라는 별도 복제 채널로 연결되어 가입한 멤버에 적용되지 않은 바이너리 로그를 적용하는 방식입니다.

### 2-1. 원격 클론 방식

- 8.0.17 버전에 도입된 클론 플러그인을 통해 모든 데이터 및 메타데이터를 일관된 스냅샷으로 가져와 재구축하는 방식입니다.
- 가입멤버, 기존 멤버간 트랜잭션 갭이 크거나, 필요 트랜잭션 일부가 기존 멤버에 없을때 원격 클론 방식을 사용합니다.
- 이때 기존 데이터는 전부 제거되고 스냅샷 데이터로 대체됩니다.
- 클론 작업 중 적용된 트랜잭션들은 바이너리 로그 복제 방식으로 한번 더 복구를 진행합니다.

## 2. 분산 복구 프로세스

- 세 단계로 나뉩니다.

### 1. 로컬 복구

가입한 적이 있는 멤버면 릴레이 로그에 적용하지 못한 트랜잭션을 먼저 적용함

### 2. 글로벌 복구

가입 멤버는 기존 멤버들에서 기증자를 선택해 해당 멤버로부터 데이터, 누락 트랜잭션을 가져와 적용합니다. 이때 현재 그룹에서 처리되는 트랜잭션들은 내부적으로 캐싱합니다.

### 3. 캐시 트랜잭션 적용

캐싱해서 보관하던 트랜잭션을 적용해 최종적으로 그룹에 참여합니다.

- 그룹에 새로운 멤버가 가입됐을때

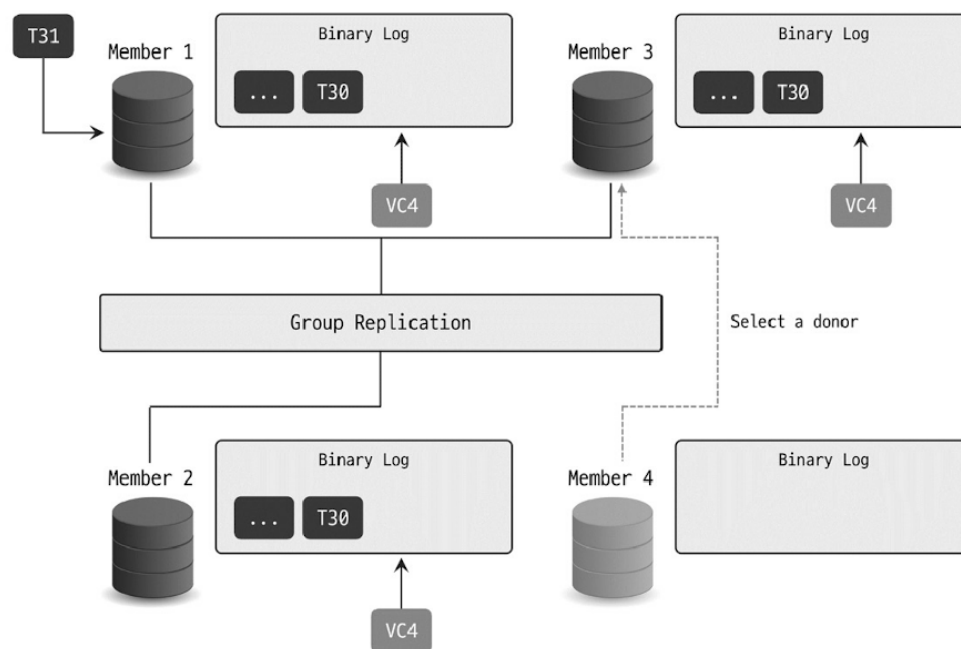


그림 17.15 그룹에 새로운 멤버 가입

#### ◦ 바이너리 로그 복제 방식에선

- 가입 멤버가 그룹에 참여한 시점까지만 복구 작업을 진행함, 복구 작업동안 그룹에서 처리된 트랜잭션들은 캐싱합니다.
  - 가입 멤버가 그룹에 참여했을때 해당되는 뷰 변경 로그 이벤트를 만나면 복제 중지하고 캐싱된 트랜잭션을 적용합니다.
- 가입한 멤버에서 분산 복구 작업을 진행

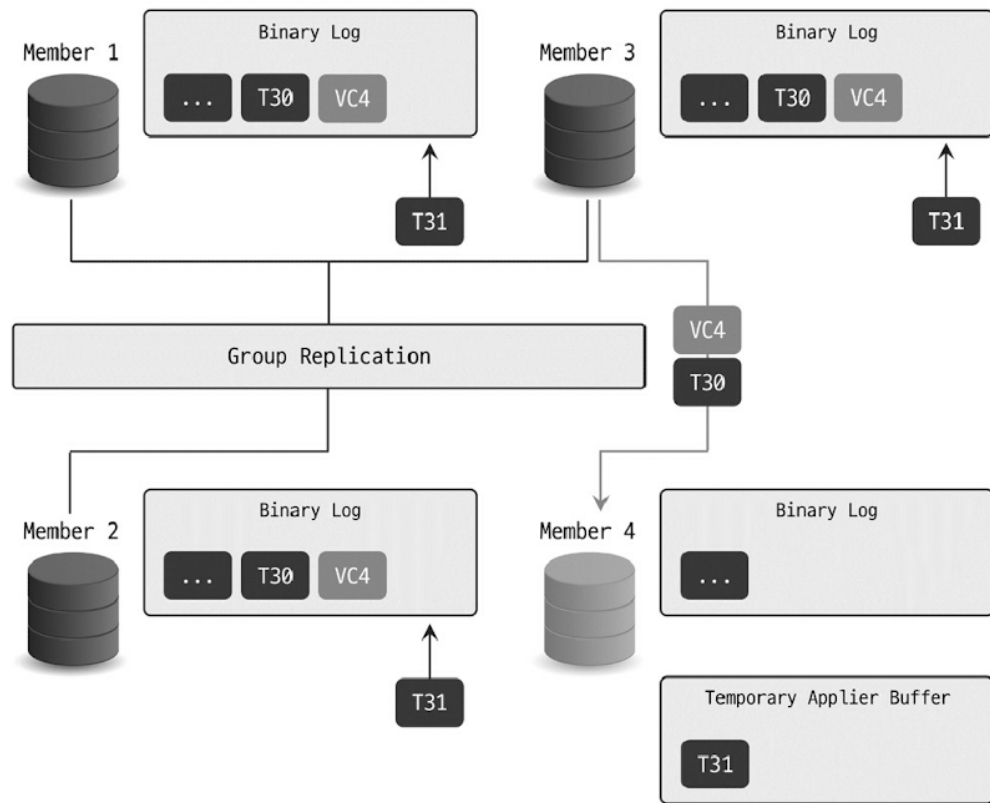


그림 17.16 가입한 멤버에서 분산 복구 작업을 진행

- 분산 복구 작업 완료

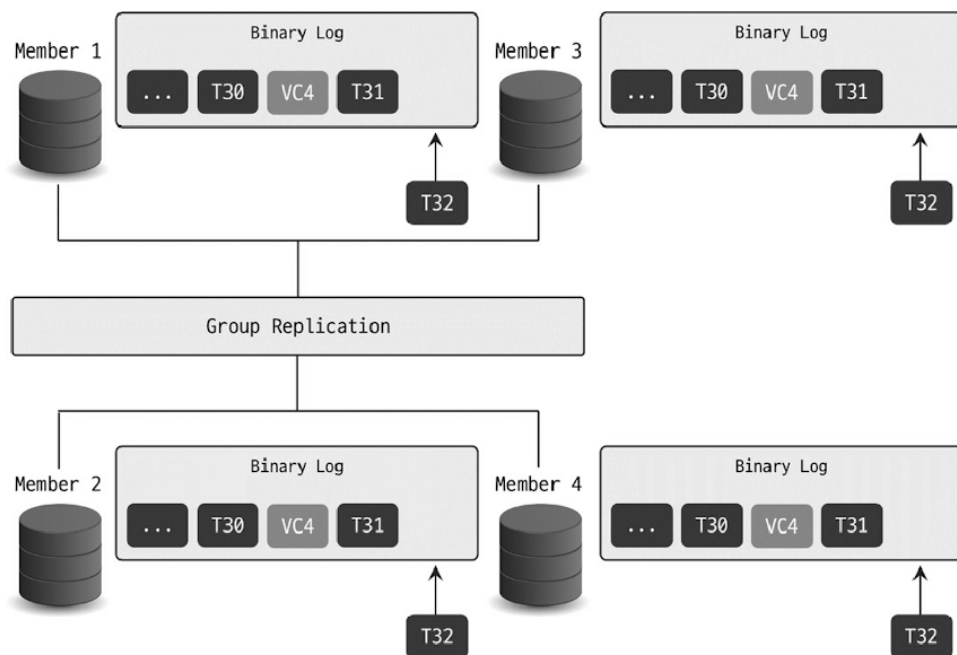


그림 17.17 분산 복구 작업을 완료해서 그룹의 공식적인 멤버가 된 가입 멤버

- 가입 멤버는 그룹 뷰 상태가 ONLINE이 됨

### 3. 분산 복구 설정

- 사용자는 분산 복구에서 아래 설정들을 시스템 변수를 통해 필요에 맞게 설정할 수 있습니다.
    - 연결 시도 횟수
    - 연결 시도 간격
    - 가입한 멤버를 ONLINE 상태로 표기하는 시점
- 다만 적용후 바로 적용되지 않고 그룹 복제 중지 및 재시작을 해주어야 합니다.

### 4. 분산 복구 오류 처리

- 다음과 같은 분산 복구 작업 중 오류가 발생하면 자동으로 다시 작업을 시작하는 장애 메커니즘이 구현돼 있습니다.
  - 기증자로 선택한 그룹 멤버로의 연결이 인증 문제 등으로 인해 정상적으로 이뤄지지 않는 경우
  - 바이너리 로그 복제 방식으로 복구 작업을 진행하는 중에 레플리케이션 I/O 스레드 또는 SQL 스레드에서 에러가 발생한 경우
  - 원격 클론 작업이 실패하거나 혹은 완료되기 전에 중단된 경우
  - 복구 작업 동안 기증자 멤버에서 그룹 복제가 중단된 경우
- 아래 경우에선 더이상 분산 복구 프로세스가 진행될 수 없으며 가입 멤버는 그룹을 떠납니다.
  - 가입 멤버가 group\_replication\_recovery\_retry\_count 시스템 변수에 지정된 재시도 횟수를 모두 소진한 경우
  - 가입 멤버에 필요한 트랜잭션이 그룹 멤버들의 바이너리 로그에 존재하지 않으며, 원격 클론 방식으로 복구 작업을 진행할 수 없는 경우
  - 가입 멤버가 그룹에서는 존재하지 않는 트랜잭션을 가지고 있는 상태에서 바이너리 로그 복제 방식으로 복구 작업이 진행되는 경우
  - 가입 멤버가 전체 그룹 멤버에 대해 원격 클론 방식과 바이너리 로그 복제 방식을 모두 시도해봤지만 전부 실패해서 더이상 시도해 볼 멤버가 존재하지 않는 경우
  - 복구 작업이 진행되는 중에 가입 멤버에서 그룹 복제가 중단된 경우

---

## 17.2.7~8 그룹 복제 요구사항 및 그룹 복제 제약 사항

필요할때 595 ~ 598 page를 참조

---

### ✓ 17.3 MySQL 셸

- MySQL을 위한 고급 클라이언트 툴로, 단순히 SQL문 실행만 가능했던 기존 클라이언트 툴인 mysql보다 좀 더 확장된 기능들을 제공함
- SQL 뿐만 아니라 JS, Python 언어 모드도 지원함
- MySQL 서버에 대해 쉽고 편리하게 작업할 수 있는 API를 제공함
  - **X DevAPI** : X프로토콜을 사용해 MySQL 서버에서 관계형 데이터와 문서 기반 데이터를 모두 처리할 수 있게 하는 API
  - **AdminAPI** : MySQL 서버의 설정을 변경하고 InnoDB 클러스터 및 InnoDB 레플리카셋을 구축할 수 있게 하는 API
- 사용자는 글로벌 객체들과 각 객체에 구현돼 있는 메서드를 통해 API를 사용합니다.
- 개중에 아래 글로벌 객체는 **JS** 및 **Python** 모드에서만 사용 가능합니다.
  - **session**, **dba**, **cluster**, **rs**, **db**, **shell**, **util**  
(자세한 설명은 599page 참조)

InnoDB 레플리카셋은 8.0.19부터 도입된 기능, 내부적으로 그룹 복제가 아니라 mysql 기본 복제를 사용하지만, mysql 셸에서 InnoDB 클러스터와 동일하게 Admin API를 사용해 복제를 관리할 수 있습니다.

기본 복제에서 사용한 기술들을 기반으로 하고 있습니다.