ORACLE

Controlled Deletion of Obsolete UUIDs from GTID State

Safe, persistent cleanup for retired GTID identities

Nuno Carvalho

Replication Lead, Oracle

May 26, 2026

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, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

Goals for this discussion

The feature is still the topic; the purpose is collaboration.

Get feedback on the feature direction before implementation details harden.

- Validate the operational problem with people running large or long-lived GTID topologies.
- Identify tooling, backup, failover, and restore assumptions that may be affected.
- Get contribution and collaboration on semantics and implementation.

Problem statement

Obsolete UUIDs accumulate forever, even when the server identity is gone.

What grows

- gtid_executed and gtid_purged keep historical server identities.
- RECEIVED_GTID_SET can retain UUIDs from retired sources.
- External automation keeps comparing larger and noisier GTID sets.

Problematic examples

- A fleet replaces hosts frequently; each replacement leaves another UUID behind.
- A restored or cloned server is later retired, but its GTIDs remain in topology checks.
- Incident response must inspect GTID sets dominated by identities that are no longer operationally relevant.

Requirements

Cleanup must be explicit, safe, persistent, and online.

Safety

- New syntax with validation and a dedicated privilege.
- Reject removal when the UUID/TSID is still in use locally.
- Crash-safe recovery to a consistent GTID state.

Topology readiness

- Pre-check whether the operation can succeed on all nodes.
- Expose current and historical deletion state for observability.
- Audit GTID-changing statements for later troubleshooting.

Online behavior

- Replication, Group Replication, backups, and PITR remain valid.
- GTID comparisons keep defined semantics during cleanup.
- No expected performance regression.

Challenges

GTID state is not just metadata; many components depend on comparisons over time.

Discontinuity

- A UUID disappearing instantly can make old and new GTID snapshots incomparable.
- Session consistency and WAIT_FOR_EXECUTED_GTID_SET must not hang or lie.
- Failover and replica-selection logic must keep comparing progress correctly.

Timing

- Old UUID transactions must be replicated and purged where required.
- The delete marker must itself replicate and be consumed before final removal.
- Backups and external tools may hold older GTID snapshots longer than the server knows.

Compatibility

- GTID set operations need tombstone-aware semantics.
- Binary GTID encoding needs a format that can represent tombstones.
- MySQL Shell and third-party tools may need semantic updates.

Examples of non-solutions

These approaches reduce metadata, but breach the online or correctness requirements.

One-shot deletion

- Remove the UUID from gtid_executed and gtid_purged in one transaction.
- Problem: older snapshots, backups, replicas, and tools may still compare against the removed intervals.
- Result: GTID subset/equality checks can become misleading around the deletion point.

Offline reset procedure

- Stop writes, wait for every node, stop replication, reset GTIDs everywhere, then restart.
- Problem: this already exists operationally, but it is disruptive and rarely acceptable at scale.
- Result: operators keep carrying obsolete UUIDs instead of doing regular cleanup.

Benefits

Bounded GTID metadata

Retired UUIDs stop accumulating indefinitely in long-lived deployments.

Clearer operations

Failover, diagnostics, backups, restores, and provisioning work with less noise.

Supported cleanup

Ad hoc resets become an auditable, privilege-protected, online server feature.


ORACLE