Timeline and lsn

Timeline ID (TLI), WAL File, and LSN in PostgreSQL (With Examples)

What is a Timeline ID in PostgreSQL?

A **Timeline ID (TLI)** is a unique identifier that tracks different versions of a PostgreSQL database's history. It ensures that standby servers follow the correct WAL (Write-Ahead Log) sequence.

- **TLI increases when a failover or promotion occurs.**
- It prevents conflicts if multiple servers generate WAL files after a failover.

⍰ Example of Timeline Change:

If a PostgreSQL primary crashes and a standby is promoted, the new primary **increments the timeline ID**, indicating a new branch in WAL history.

Relation Between Timeline ID, WAL File, and LSN

Log Sequence Number (LSN)

- The **LSN** is a unique position in the WAL.
- It marks the exact byte location where a transaction is written.
- Example LSN: 0/16B4D0A8

Write-Ahead Log (WAL) Files

- WAL files store transaction logs and follow a strict sequence.
- Each WAL file name contains:
  - **Timeline ID**
  - **LSN (Log Sequence Number) range**

000000020000000300000000A

Breakdown:-

- 00000002 → Timeline ID (2)
- 00000003 → Log file segment
- 0000000A → WAL segment number

5. How Timeline ID and LSN work together

Timeline ID = which branch/version of the database

LSN = exact position of a change in that branch

Together they tell PostgreSQL:

“Read Timeline 2, starting from LSN 0/4000 to get changes after failover.”

Timeline ID (TLI) in WAL File Naming

- When a timeline change occurs, new WAL files **start with a new timeline ID**

Example WAL file names before and after a failover:-

```
00000001000000030000000A  # Timeline 1 (Old primary)
00000002000000030000000B  # Timeline 2 (New primary)
```

⍰ Example: How Timeline ID Changes

Initial setup (Primary = Timeline 1)

- WAL file: 00000001000000030000000A
- LSN: 3/A000000

Primary crashes, standby is promoted

- New WAL file: 00000002000000030000000B
- New LSN: 3/B000000
- New Timeline ID: 2

Old primary tries to reconnect (Timeline mismatch!)

- The standby is ahead (TLI=2), but the old primary is still on TLI=1, causing replication to fail.

Fixing Timeline ID Mismatch on a Standby Server

If a standby server’s timeline ID differs from the primary, it **cannot continue replication**. Here’s how to fix it:

⍰ Method 1: Fix Using pg_rewind (Preferred)

✓Use if the standby was previously promoted and is now being reattached.

- **Stop the standby:**

```
systemctl stop postgresql
```

- **Run** pg_rewind **to sync timelines:**

```
 pg_rewind --target-pgdata=/var/lib/pgsql/15/data --source-
server="host=primary_ip user=postgres" -P
```

- -D /var/lib/pgsql/15/data → Path to the old primary's data directory.
- --source-server="host=primary_ip user=postgres port=5432" → Connects to the new primary.
- -P → Shows progress.

- **Restart the standby:**

```
 systemctl start postgresql
```

⍰ Key Takeaways

✓Timeline ID (TLI) tracks PostgreSQL history after failover.
✓WAL files store transaction logs and contain the TLI to prevent conflicts.
✓LSN ensures sequential WAL application.
✓If Timeline ID mismatch occurs on a standby:

- Use pg_rewind if the standby was promoted earlier.
- Use pg_basebackup if pg_rewind is not possible.

===
=

1. WAL file name

Example:

00000001000000030000000A

Breakdown:

Part	Meaning
00000001	Timeline ID = 1
00000003	Log file number (XLOG file number)
0000000A	Segment number inside that log file

WAL file number = 3 → This is the big block in the timeline.

Segment number = 10 (0A in hex) → The 10th segment in this log file

WAL file names count segments sequentially: PostgreSQL increments them as WAL fills up.

2. LSN format

LSN example:

3/A000000

Breakdown:

Part	Meaning
3	WAL file number (same as XLOG number)
A000000	Byte offset inside that WAL file

3 → means it's in WAL file number 3

A000000 → how many bytes from the start of WAL file 3

So the 3 in the LSN corresponds to the WAL file number in the WAL filename.

3. How it works together

WAL file: 00000001000000030000000A

Timeline = 1

WAL file number = 3

Segment = 10

LSN 3/A000000

WAL file number = 3 → points to same WAL file

Offset A000000 → points inside that WAL file

In simple words:

WAL filename = the big container

LSN = exact position inside that container

4. Analogy

WAL file = page in a book

Segment = paragraph on that page

LSN = line number inside that paragraph

So 3/A000000 → WAL page 3, line far inside (byte offset A000000)