What Are the Different Types of Files Stored in PostgreSQL's pg_wal Directory?

The **`pg_wal`** directory in PostgreSQL (formerly known as **`pg_xlog`** in versions prior to **PostgreSQL 10**) is crucial for the database's Write-Ahead Logging (**WAL**) system. This directory stores various files that ensure the durability, consistency, and recovery capabilities of the database. Below is an overview of the types of files found inside the **`pg_wal`** directory and their use cases:

1. WAL Segment Files

- Naming Convention:WAL segment files follow a specific naming convention, such as **`000000010000000000000066`**. This name consists of a **timeline ID**, **log file ID**, and **segment ID**, **all in hexadecimal format**.

Use Case:

— These files store the actual WAL records, which log every change made to the database. WAL segment files are essential for ensuring that all transactions can be replayed during crash recovery, point-in-time recovery (PITR), or during replication to standby servers.

— Each WAL segment file is typically **16MB** in size by default (this size can be configured).

- Example File:**`000000010000000000000066`**

2. Partial WAL Files (`.partial`)

Naming Convention: These files are named similarly to WAL segment files but with a **`.partial`** extension, such as **`000000010000000000000066.partial`**.

Use Case:

— A **`.partial`** file is created when a **WAL** segment is not fully written, typically due to an unexpected shutdown or crash. This file indicates that the WAL segment is incomplete.

— During recovery, PostgreSQL will attempt to complete these files if possible, or they may be discarded if deemed unrecoverable.

- Example File:**`000000010000000000000066.partial`**

3. Timeline History Files (`.history`)

Naming Convention: Timeline history files are named using the format `0000000B.history`, where the number corresponds to the timeline ID.

Use Case:

— These files track the history of timeline changes. A timeline change occurs, for instance, when a standby server is promoted to primary, or during point-in-time recovery (PITR) when a specific recovery target is reached.

— The **`.history`** file records the LSN where the timeline diverged and provides a record of the sequence of events that led to the creation of the new timeline.

Example File:**`0000000B.history`**

4. Status Files Inside `archive_status` Directory

- **Location:** These files are located in the `archive_status` subdirectory within **`pg_wal`**.

  File Types:

  `.ready` Files:

  Naming Convention: These files are named similarly to WAL segment files but with a `.ready` extension, such as **`000000010000000000000066.ready`**.

  Use Case: A **`.ready`** file indicates that the corresponding WAL segment file is ready to be archived. PostgreSQL will attempt to archive the WAL segment by executing the **`archive_command`**.

- **`.done` Files:**

  Naming Convention: These files are named similarly to WAL segment files but with a `.done` extension, such as **`000000010000000000000066.done`**.

  Use Case: A **`.done`** file indicates that the corresponding WAL segment has been successfully archived, and PostgreSQL can safely recycle or remove the WAL file.

  - Example Files:**`000000010000000000000066.ready`**, **`000000010000000000000066.done`**

5. Temporary Files (`.tmp`)

Naming Convention: Temporary files inside `**pg_wal**` may have various extensions, such as `**.tmp**`.

Use Case:

— These files are created during the process of writing, archiving, or transferring WAL segments. They represent intermediate states and are typically renamed or deleted once the operation is complete.

— For example, a temporary file might be used when a WAL segment is being copied to an archive location.

- Example File:`**000000010000000000000066.tmp**`

6. Backup Label Files (`backup_label` and `backup_label.old`)

Naming Convention: The file is named `**backup_label**`, and an old version of it may be retained as `**backup_label.old**`.

Use Case:

— The `**backup_label**` file is created when a base backup is taken using tools like `**pg_basebackup**`. It contains information about the backup, including the starting **LSN** of the backup and the timeline.

— This file is crucial during recovery to identify the starting point of **WAL** replay.

Example File: `backup_label`

7. End-of-Recovery Markers (`recovery.signal`, `standby.signal`)

Naming Convention: These files are named `**recovery.signal**` and `standby.signal`.

Use Case:

— `**recovery.signal**`: Indicates that the database should start in recovery mode. This file is typically created when you perform point-in-time recovery (**PITR**).

— `**standby.signal**`: Indicates that the database should start in standby mode, typically used in replication setups to keep the server as a hot standby.

- Example Files: `**recovery.signal**`, `**standby.signal**`

Summary of Use Cases

WAL Segment Files: Store WAL records crucial for database recovery, replication, and point-in-time recovery.

Partial Files: Represent incomplete WAL segments, typically created during crashes.

History Files: Track timeline changes, important for understanding and managing recovery across different timelines.

Status Files (`archive_status`): Manage the archiving process of WAL segments, indicating whether they are ready for or have completed archiving.

Temporary Files: Used during the WAL processing stages, often transient and removed once the operation completes.

Backup Label Files: Essential for identifying the start of WAL replay during recovery from a base backup.

End-of-Recovery Markers: Control the startup mode of the PostgreSQL instance, determining whether it should start in recovery or standby mode.

Conclusion

The `**pg_wal**` directory is a vital part of PostgreSQL's WAL system, housing a variety of files that serve different roles in ensuring database durability, consistency, and recoverability. Understanding the purpose of each file type helps database administrators manage the PostgreSQL environment more effectively, especially in scenarios involving replication, backup, and recovery.