8. High Availability and Replication.

Introduction to PostgreSQL

HIGH AVAILABILITY

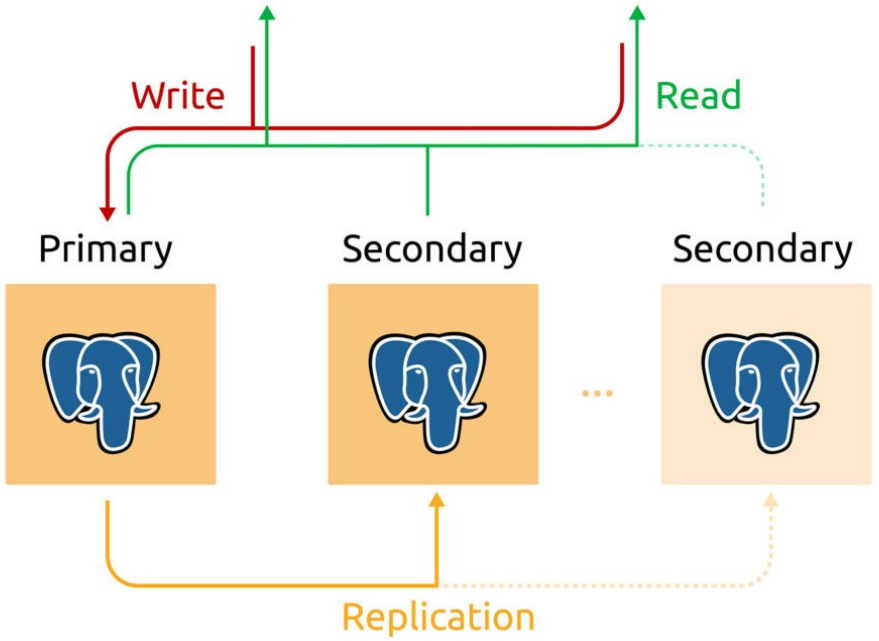
- High Availability
  - Minimizing downtime
  - Ensuring that the database service remains accessible even during failures.
  - Often related to how a cluster hands loads at peak times
- Failover and Replication
  - Replication ensures data is copied to multiple instances
  - Failover automatically switches to a standby instance if the primary fails.
  - Automated failover tools ike Patroni, repmgr, PgPool-II, and PostgreSQL Automatic Failover (PAF) can automate failover and manage the high availability setup
- Load Balancing:
  - Balancing read queries across replicas helps distribute the load, enhancing performance and availability.
  - The highest demands on the system tend to be read requests.

REPLICATION

- Replication
  - Copying data from one database (primary) to others (replicas).
- Streaming Replication:
  - Uses WAL (Write-Ahead Logging) to stream changes from the primary to replicas in near real-time.
  - Replicas can be set up as hot standbys (readable) or warm standbys (not readable until promoted).
- Log Shipping:
  - The primary server periodically ships completed WAL files to the standby servers, which apply them to their copy of the database

REPLICATION

- Logical Replication:
  - Replicates data changes based on logical changes (like inserts, updates, deletes)
  - Allows selective data replication, supports bi-directional replication, and is useful for replicating only specific tables or data subsets.
- Synchronous Replication:
  - Ensures that transactions are committed to both the primary and at least one standby before being acknowledged to the client.
- Asynchronous Replication:
  - Primary does not wait for the replica's acknowledgment



PLANNING

- Create the primary and standby servers so that they are as similar as possible, at least from the perspective of the database server.
  - Path names associated with tablespaces will be passed across unmodified, so both primary and standby servers must have the same mount paths for tablespaces.
  - Hardware need not be exactly the same
  - Experience shows that maintaining two identical systems is easier than maintaining two dissimilar ones over the lifetime of the application and system.
  - The hardware architecture must be the same replicating a 32-bit system to a 64-bit system will not work.
- The servers should be running the same major version
  - WAL log shipping between servers running different major releases is not possible.
  - Different minor release versions probably will work, but should be avoided

STANDBY SERVER OPERATION

- A server is in standby mode if a standby.signal file exists in the data directory when the server is started.
- In standby mode:
  - The server continuously applies WALs received from the primary server
  - Standby mode is exited and the server switches to normal operation when pg_ctl promote is run, or pg_promote() is called.
  - Before failover, any WAL immediately available in the archive or in pg_wal will be restored, but no attempt is made to connect to the primary.

SETTING UP REPLICATION

- Configuration on the Primary Server
  - Edit postgresql.conf:
    - *Set wal_level to replica or logical (for logical replication).*
    - *Set max_wal_senders to the number of standby servers*
    - *Set archive_mode to on and configure archive_command if using log shipping.*
    - *Configure listen_addresses to allow connections from standby servers.*
  - Edit pg_hba.conf:
    - *Add entries to allow replication connections from the standby servers.*
- Configuration on the Standby Server
  - Base Backup:
    - *Use pg_basebackup or other methods to create a copy of the primary server's data directory.*
    - *Edit postgresql.conf with settings like primary_conninfo, pointing to the primary server.*
    - *primary_conninfo specifies the connection details for the primary server, including host, port, user, and password*

REPLICATION PROCESS

- Once configured, the standby server connects to the primary server and starts streaming WAL records.
- The standby server replays these records to keep its copy of the database updated with the primary server.
- In streaming replication, this process is continuous, with the standby receiving changes as they occur on the primary.

FAILOVER

- In case of a primary server failure
  - A standby can be promoted to act as the new primary.
  - This can be done manually or automatically using tools like pg_auto_failover or Patroni.
- Switchover
  - A planned switch where the roles of primary and standby are reversed, usually for maintenance or load balancing.

MONITORING REPLICATION

- Several tools and views for monitoring replication,
  - pg_stat_replication on the primary server
  - pg_stat_wal_receiver on standby servers.
  - These views provide information on replication status, lag, and connection details.

MONITORING REPLICATION

- Several tools and views for monitoring replication,
  - pg_stat_replication on the primary server
  - pg_stat_wal_receiver on standby servers.
  - These views provide information on replication status, lag, and connection details.
- Handling Replication Lag
  - Replication lag can occur due to network issues, slow disk I/O, or high write activity on the primary.
  - Monitor and tune parameters like max_standby_streaming_delay to manage how standbys handle replication delays.
- CAP Theorem
  - Find the balance of consistency and availability needed for specific use cases

End Module