Key PostgreSQL Concepts for SQL Server DBAs

Prepared by: Dave Bland Manager of Database Engineering — Lexia Learning @SQLDAVE29 daveb8782@gmail.com www.davebland.com LinkedIn: linkedin.com/in/daveblanddba

1. PostgreSQL Architecture Overview

PostgreSQL uses a **cluster-based architecture**, which differs significantly from SQL Server's instance model.

A PostgreSQL cluster contains:

- A single set of configuration files
- Multiple databases
- Shared roles and permissions
- Shared background processes

Roles, settings, and extensions apply at the **cluster level**, not per database.

2. MVCC: The Heart of PostgreSQL Concurrency

PostgreSQL uses **Multi-Version Concurrency Control (MVCC)** to ensure readers never block writers.

Key points:

- Every update creates a new tuple version
- Old versions remain until vacuumed
- Long-running transactions prevent cleanup
- Index-only scans depend on visibility map accuracy

MVCC is the biggest conceptual shift for SQL Server DBAs.

3. Vacuum & Autovacuum

Vacuum is essential for PostgreSQL health.

Vacuum:

- Removes dead tuples
- Prevents table bloat

- Maintains visibility maps
- Supports index-only scans

Autovacuum must be tuned for:

- High-update tables
- Large tables
- Long-running transactions
- High-throughput OLTP workloads

4. WAL: Write-Ahead Logging

PostgreSQL's **Write-Ahead Logging (WAL)** is similar to SQL Server's transaction log but behaves differently:

- WAL is always sequential
- WAL archiving enables point-in-time recovery (PITR)
- WAL segments are fixed-size (16MB default)
- Tools like pgBackRest and WAL-G rely heavily on WAL

Understanding WAL is critical for backup, recovery, and replication.

5. Extensions: PostgreSQL's Superpower

PostgreSQL supports a rich ecosystem of extensions:

- **pg_stat_statements** — query performance insights
- **PostGIS** — geospatial capabilities
- **pg_cron** — scheduling inside PostgreSQL
- **hstore / jsonb** — semi-structured data
- **citext** — case-insensitive text

Extensions allow PostgreSQL to be customized far beyond SQL Server's feature set.

6. Configuration & Performance Tuning

PostgreSQL relies heavily on configuration tuning.

Key settings:

- shared_buffers
- work_mem
- maintenance_work_mem
- effective_cache_size
- wal_level
- max_wal_size

PostgreSQL does not auto-tune memory or parallelism.

7. Query Planning & Statistics

PostgreSQL's planner is cost-based and relies on:

- Table statistics
- Histogram data
- Correlation values
- MCV (most common values) lists

Statistics are updated via ANALYZE or autovacuum.

Prepared statements behave differently than SQL Server's parameter sniffing model.

8. Replication & High Availability

PostgreSQL supports:

- Streaming replication
- Logical replication
- Physical replication
- Cascading replication
- Synchronous or asynchronous modes

Failover is typically handled by:

- Patroni
- repmgr
- pg_auto_failover

9. Backup & Recovery Essentials

SQL Server DBAs must understand the two backup types:

- **Logical** — pg_dump, pg_dumpall
- **Physical** — pg_basebackup + WAL

Production systems require physical backups + WAL archiving.

10. Key Differences Summary

Concept	SQL Server	PostgreSQL
Concurrency	Readers block writers	MVCC (readers never block)
Cleanup	Ghost cleanup	Vacuum/autovacuum
Log	Transaction log	WAL segments
Scheduler	SQL Agent	cron/pgAgent/pg_cron
Extensions	Limited	Extensive ecosystem
Architecture	Instance	Cluster
Statistics	Auto-maintained	ANALYZE + autovacuum

11. Final Takeaways

- MVCC changes how you think about blocking and deadlocks
- Vacuum tuning is essential for performance
- WAL drives backup, recovery, and replication
- Extensions unlock powerful capabilities
- PostgreSQL requires more configuration tuning than SQL Server