

PostgreSQL Deep Dive

Internals, Performance & Practical Mastery - 2 days

FA-0015 · 2 days

Data & Analytics · DevOps, Cloud & Infrastructure

Introduction

PostgreSQL is incredibly powerful, but real performance and reliability come from understanding what happens under the hood. Simply writing SQL isn't enough when your dashboards slow down or production experiences latency, you need to *think like PostgreSQL*.

This 2-day course demystifies core internals (tuples, pages, WAL, plans), equips you with optimization techniques that matter in practice, and exposes you to powerful extensions and time-series management strategies (including TimescaleDB compression). You will learn how PostgreSQL executes queries, why designs affect performance, how to benchmark and debug, how to tune networking and connections, and how to leverage PostgreSQL's ecosystem for development efficiency.

This course is designed for people who already:

- Know SQL
 - Have used PostgreSQL in real applications
 - Can navigate a Linux server and work with terminal/SSH
 - Are ready for advanced debugging, performance, and architectural insights
-

Learning outcomes

By the end of this course, learners will be able to:

- Explain PostgreSQL storage internals and relate them to performance and debugging
- Analyze and optimize query plans using EXPLAIN ANALYZE, understand plan choices
- Identify and apply the right index and storage strategies for complex workloads
- Benchmark and profile PostgreSQL to diagnose bottlenecks

- Use PostgreSQL extensions that accelerate development and debugging
 - Apply TimescaleDB effectively, including compression and chunking strategies
 - Write efficient analytical queries for large datasets including inspection/time-series data
 - Use PostgreSQL features like backup/restore, scheduling, reporting, and more
 - Configure connections, pooling, and network settings for high performance
 - Debug real application scenarios using PostgreSQL tools and logs
-

Prerequisites

Learners should have:

- Strong foundation in SQL (joins, aggregations, subqueries)
 - Familiarity with PostgreSQL basics
 - Linux command-line experience
 - VS Code installed and configured to connect to PostgreSQL
 - Access to a remote PostgreSQL server (SSH access via port 22)
 - No restriction on PostgreSQL default port 5432
 - Basic understanding of database concepts (tables, indexes, normalization)
-

Training outline

1. 1. PostgreSQL Architecture & Storage Internals

PostgreSQL Process Model

- Server processes (postmaster, bgworkers, autovacuum)
- Backend lifecycle

Memory components

- Shared buffers
- Work mem / maintenance work mem

Storage unit fundamentals

- Pages/blocks
- Tuples and item identifiers
- Visibility map, Free space map

Write-Ahead Logging (WAL)

- Logging mechanism

- Checkpoints and crash recovery

Vacuum and Autovacuum

- Tuple visibility
 - Bloat, freeze, and transaction IDs
 - Practical: Inspecting physical files with tools (pg_filedump or similar)
-

2. 2. Query Processing and Planner Internals

- SQL parse → rewrite → plan → execute workflow

Statistics and the planner

- How stats influence plan selection
- ANALYZE, VACUUM ANALYZE

Plan types and join algorithms

- Nested loop
- Hash join
- Merge join

Cost model basics

- I/O cost vs CPU cost
 - Planning time vs execution time
 - Query plan caching and invalidation
 - Practical: Read and interpret complex plans
-

3. 3. Advanced EXPLAIN ANALYZE Techniques

- EXPLAIN vs EXPLAIN ANALYZE

Buffer and timing output

- Shared hit/read/write statistics
- Local vs temp buffers

Identifying common performance issues

- Seq scan vs Index scan
- Bitmap heap scans explained
- Using auto_explain to log slow plans

Tools that visualize plans

- External and VS Code extensions (pgAdmin explain viewer alternatives)
-

4. 4. Indexing Strategies and Storage Optimization

Index types and when to use them

- B-tree
- Hash indexes
- GIN, GiST, SP-GiST
- BRIN
- Expression and partial indexes
- Covering indexes and index only scans
- Multi-column index ordering and pitfalls
- Index impact on write performance

Table partitioning

- Range vs list
- Declarative partitioning internals

Storage parameters

- Fillfactor, toast thresholds
 - Practical: Design index strategy for large datasets
-

5. 5. Benchmarking & Performance Tuning

Establishing performance baselines

- pgbench usage
- Custom workloads

Load profiles and metrics

- TPS
- Latency percentiles

System resource monitoring

- CPU, memory, disk I/O, network

Tuning parameters for workloads

- effective_cache_size
 - random_page_cost vs seq_page_cost
 - work_mem tuning
 - Practical: Interpret and adjust based on benchmark results
-

6. 6. PostgreSQL Extensions to Accelerate Development

- Extension system and installation

Useful extensions

pg_stat_statements

- Query frequency and cost tracking
- Using for performance auditing

auto_explain

- Logging slow queries automatically

pg_repack

- Online reorganization of tables/indexes

hypopg

- Hypothetical indexes for planning

- pg_trgm / fuzzy search
- citext, uuid-osp, tablefunc

- Extension best practices and caveats
 - Practical: Hands-on with each extension
-

7. 7. Time-Series with TimescaleDB

Why TimescaleDB

- Chunking and time partitioning strategy

- Hypertable creation and configuration

Compression policies

- Columnar compression principles
- Policy automation

Query speed against compressed data

- Indexing compressed chunks
- Using real-time vs historical chunks

Continuous aggregates

- Refresh strategies
- Use cases for dashboards

- Practical: Building a time-series ingestion and query workflow
-

8. 8. Advanced Query Best Practices

SQL query anti-patterns

- Overuse of subqueries vs CTEs
- Unnecessary sorting and functions on indexed columns
- Window functions and performance considerations
- Set operations vs joins
- Aggregation strategies for large datasets
- Query rewrite and refactoring techniques

Tips for analytics dashboards

- Pre-aggregations
 - Materialized views
 - Practical: Transform slow queries to efficient designs
-

9. 9. Backup, Restore & Replication Strategies

Logical backups

- pg_dump, custom formats
- Parallel dumps

Physical backups

- pg_basebackup
- PITR (Point-in-Time Recovery)

WAL archiving

- Setting up
- Restoring from WAL

Streaming replication basics

- Synchronous vs asynchronous
 - Monitoring replication lag
 - Practical: Create a restore plan for production SLAs
-

10. 10. Connection Management & Network Optimization

PostgreSQL connection model

- Handshake and backend allocation

Connection pooling

- Why pooling matters
- PgBouncer / Pgpool basics

Tuning listen_addresses and max_connections

- Connection costs and memory footprint

TCP stack tuning basics

- Keepalives
 - Network latency considerations
 - Monitoring and diagnosing connection issues
 - Practical: Configure pool for high concurrency workloads
-

11. 11. PostgreSQL Operational Features

Scheduled jobs

- pg_cron
- External schedulers (Linux cron with scripts)

Reporting and exports

- Copy to CSV/JSON
- ETL considerations

Logging and audit trails

- Log levels
- Integrating with log aggregators

Security practices

- Roles and privileges
 - SSL/TLS basics for remote clients
 - Practical: Configure production-ready logging and maintenance
-

12. 12. Capstone Problem Sets*

Real performance case study

- Diagnose slow system with logs and explain plans
 - Apply fixes and measure
 - Time-series ingestion and dashboard query
 - Index redesign exercise
 - Compression policy evaluation
-

Disclaimer

* The course outline provided represents the intended scope and depth of the training. However, all topics, sequencing, and coverage may be adjusted at the discretion of the trainer to ensure the most effective learning experience. Modifications may be necessary due to participant skill variations, environment constraints, firewall or network limitations, infrastructure accessibility, time availability, or other technical considerations. The instructor reserves the right to prioritize, expand, condense, or substitute topics based on real-time assessment of class progress and practical feasibility, while maintaining the overall objectives and learning outcomes of the course.

© 2026 Faysal Aziz · Kuala Lumpur

<https://faysalaziz.com/courses/postgresql-deep-dive/>