CSE 3523 & CSE 3523S

Database Management Systems

Semester: Spring 2026

Audience: Undergraduate Computer Science Students

Prerequisite: CSE 1208 (Programming Language and Application II)

Instructor: Sohaib Abdullah, Assistant Professor, Department of CSE

---

1. Course Description

Data is the most critical asset of any modern organization. This course moves beyond the traditional view of databases as simple storage systems and treats them as **core infrastructure components** that underpin modern software platforms, data-driven services, and distributed applications.

While the course covers the foundational theory of the **Relational Model**, it is deliberately designed to be **practical, systems-oriented, and industry-aligned**. Students will work with an open-source technology stack widely used in modern software engineering environments and learn how databases are designed, optimized, secured, and scaled in real systems.

Students will learn to design schemas that enforce integrity, write expressive and performant SQL, understand how database engines execute and optimize queries, and reason about engineering trade-offs involving consistency, availability, performance, and fault tolerance.

---

2. Course Objectives

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

- **Master Modern SQL**

Write complex, correct, and efficient SQL queries using joins, subqueries, common table expressions (CTEs), window functions, and semi-structured data (JSON).

- **Design Robust and Scalable Schemas**

Apply ER modeling and normalization theory to design schemas that enforce integrity, while understanding when and why controlled denormalization is used in practice.

- **Understand Database Internals**

Explain how data is stored, indexed, retrieved, and optimized, including B-Tree indexes, query execution strategies, and cost-based optimization.

- **Engineer for Concurrency and Reliability**

Understand ACID properties, isolation levels, multi-version concurrency control (MVCC), logging, crash recovery, and failure handling.

- **Build and Operate Real Systems**

Use Docker, migrations, and automated testing pipelines to deploy, evolve, and validate database-backed applications.

---

3. Methodology & Tools

This course adopts a **systems-first, infrastructure-as-code approach**.

- **Primary Database Engine:** PostgreSQL
- **Development Environment:** Docker and Docker Compose
- **Version Control & Automation:** GitHub Classroom, GitHub Actions
- **Application Layer:** Python (FastAPI, SQLAlchemy)
- **Schema Migration:** Alembic
- **Client Tools:** psql, DBeaver or TablePlus

---

4. Reference Materials

[S] Academic Core

- Silberschatz, Korth, Sudarshan – *Database System Concepts*, 7th Edition

[K] Systems & Industry Perspective

- Kleppmann, Riccomini – *Designing Data-Intensive Applications*, 2nd Edition

[AP] Practitioner Reference

- Dimitri Fontaine – *The Art of PostgreSQL*

Supplementary materials include PostgreSQL official documentation and selected lecture notes from CMU 15-445.

---

5. Course Overview

Module	Weeks	Theory Focus (CSE 3523)	Lab Focus (CSE 3523S)
Foundations	1–3	Architecture, Relational Model, SQL (Basic → Advanced)	Docker setup, GitHub Classroom, SQL unit testing
Design & Integrity	4–6	ER Modeling, Normalization, Constraints	Capstone schema design, mock data, database programmability

Module	Weeks	Theory Focus (CSE 3523)	Lab Focus (CSE 3523S)
Internals & Performance	7–9	Storage, Indexing, Query Processing, Transactions	Application integration, migrations, performance tuning
Distributed & Modern Systems	10–12	Concurrency, Recovery, Replication, Scaling, NoSQL	Security, backup & recovery, capstone defense

---

6. Detailed Course Outline: CSE 3523 (Theory)

2 Lectures per Week (1h 20m each)

Module 1: Relational Foundations & SQL

Week 1

- L1: Database Systems Architecture. File Systems vs DBMS. Storage vs Compute
- L2: Data Models and the Relational Model. Relations, Tuples, Keys, Schema vs Instance

Week 2

- L3: Relational Algebra and Mapping to SQL
- L4: SQL as a Declarative Language. Logical Query Processing, Aggregation

Week 3

- L5: Joins, Subqueries, Common Table Expressions (CTEs)
- L6: Window Functions, Ranking, JSON and Semi-Structured Data

---

Module 2: Database Design & Integrity

Week 4

- L7: ER Modeling. Entities, Relationships, Cardinalities
- L8: Mapping ER Models to Relational Schemas

Week 5

- L9: Functional Dependencies and Normal Forms (1NF–3NF)
- L10: BCNF, Trade-offs, and Controlled Denormalization

Week 6

- L11: Constraints, Triggers, and Stored Procedures
- L12: Constraints, Triggers, and Stored Procedures

Module 3: Internals & Performance

Week 7

- L13: Storage Organization, Pages, Heap Files, Row vs Column Stores
- L14: Indexing Fundamentals: B+ Trees, Clustered vs Non-Clustered Indexes

Week 8

- L15: Index Variants, Composite Indexes, Indexing JSON Data
- L16: Query Execution Algorithms and Cost-Based Optimization

Week 9

- L17: Transactions, ACID Properties, Write-Ahead Logging
- L18: Concurrency Control, Locking Protocols, Deadlocks

Module 4: Distributed & Modern Systems

Week 10

- L19: Isolation Levels, Anomalies, MVCC
- L20: Database Security. Roles, Privileges, Row-Level Security

Week 11

- L21: Replication, Partitioning, and Consistency Trade-offs
- L22: CAP Theorem in Context and Distributed Databases

Week 12

- L23: NoSQL Systems (Document, Key-Value, Graph)
- L24: NewSQL, Time-Series Databases, Review and Synthesis

---

7. Detailed Course Outline: CSE 3523S (Lab)

1 Lab per Week (2h 20m)

- Week 1: Environment Setup. Git, GitHub Classroom, Dockerized PostgreSQL
- Week 2: SQL Boot Camp (Auto-graded)
- Week 3: Capstone Initialization. Schema Design and DDL
- Week 4: Mock Data Generation and Complex Queries
- Week 5: Advanced SQL Patterns and Analytics
- Week 6: Database Programmability (PL/pgSQL)

- Week 7: Application Integration (FastAPI + SQLAlchemy)
- Week 8: Schema Migrations with Alembic
- Week 9: Performance Tuning with EXPLAIN ANALYZE
- Week 10: Transactions and Concurrency Control
- Week 11: Security, Backup, and Recovery
- Week 12: Capstone Final Defense

---

8. Capstone Project: SmartCampus Core

Students will build and evolve a **single, production-style database schema** throughout the semester.

Theme:

High-concurrency backend for a University Management System.

Key Engineering Challenges:

- Concurrent enrollment handling using transactions and isolation levels
- Recursive prerequisite logic and schedule conflict detection
- Financial integrity with ACID guarantees
- Performance optimization using indexes and materialized views

Evaluation:

Automated testing via GitHub Actions and a final architecture and design defense.