



ЛИЦЕНЗИРОВАННЫЙ УЧЕБНЫЙ ЦЕНТР

ПРОФЕССИОНАЛЬНАЯ ПРОГРАММА · 2026

POSTGRESQL- РАЗРАБОТЧИК

от SQL до production и highload

Проектирование · Advanced SQL · PL/pgSQL

MVCC · EXPLAIN ANALYZE · Индексы · Highload

Backend integration · Docker · PgBouncer · Kafka/Outbox

150

академ. часов

6

практических проектов

1

дипломный проект

Практика на каждом этапе.

Фокус на реальной разработке и производительности.

Обновлённая программа профессии

PostgreSQL-разработчик: от SQL до production и highload

Программа переработана так, чтобы студент не просто изучал синтаксис SQL, а проходил полный инженерный путь: проектирование модели данных, сложные запросы, серверная логика, транзакции и MVCC, диагностика производительности, эксплуатация PostgreSQL и интеграция с backend-системами.

150	6	6	1
академ. часов	блоков	проектов	диплом

Цель программы

- Подготовить PostgreSQL-разработчика, который умеет проектировать структуру данных, писать и оптимизировать SQL, использовать возможности PostgreSQL осознанно и принимать решения с учётом нагрузки.
- Научить диагностировать реальные проблемы: медленные запросы, неверные планы, блокировки, рост dead tuples, ошибки индексации, неоптимальные транзакции и перегрузку соединений.
- Сформировать связь между базой данных и современной backend-архитектурой: миграции, pooling, Docker, интеграция с приложением, Outbox Pattern и событийная архитектура.

Ключевые результаты обучения

- ✓ уверенно писать SQL от базовых CRUD-операций до CTE, оконных и рекурсивных запросов;
- ✓ проектировать схемы данных, ограничения и индексы под реальные бизнес-сценарии;
- ✓ работать с JSONB, ARRAY, RANGE, views и materialized views;
- ✓ разрабатывать функции, процедуры и триггеры на PL/pgSQL;
- ✓ понимать транзакции, уровни изоляции, MVCC, блокировки и причины конфликтов;
- ✓ читать EXPLAIN ANALYZE, работать с BUFFERS и находить узкие места;
- ✓ настраивать практические элементы production-среды: connection pooling, мониторинг, backup/restore, partitioning;
- ✓ понимать, где должна жить бизнес-логика: SQL, PL/pgSQL или backend-код.

Структура программы — 150 академических часов

Блок	Содержание	Часы
1	SQL и реляционная модель	24
2	Advanced SQL и аналитические запросы	24
3	Проектирование и возможности PostgreSQL	24
4	PL/pgSQL и серверная разработка	24
5	Транзакции, MVCC и производительность	30
6	Production PostgreSQL, backend и highload	18
—	Итоговый дипломный проект и защита	6

Блок 1. SQL и реляционная модель

24 ак. часа · теория + live coding + практика + code review

1.1. PostgreSQL и реляционная модель

- Роль СУБД в приложении. Реляционная модель и понятие отношения.
- Установка и рабочее окружение: psql, pgAdmin/IDE, Docker-контейнер.
- Схемы, таблицы, типы данных, NULL, DEFAULT, IDENTITY.

1.2. DDL и целостность данных

- CREATE/ALTER/DROP. PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, NOT NULL.
- Каскадные операции и проектирование ограничений.
- Практика: модель данных по бизнес-требованиям.

1.3. SELECT и CRUD

- SELECT, INSERT, UPDATE, DELETE, RETURNING.
- Фильтрация, сортировка, агрегаты, GROUP BY/HAVING.
- INSERT ... ON CONFLICT как основа UPSERT.

1.4. JOIN и работа со связанными данными

- INNER/LEFT JOIN, self join, несколько связей.
- Многие-ко-многим, промежуточные таблицы.
- Типичные ошибки: дублирование строк и неверные агрегаты.

ПРАКТИЧЕСКИЙ ПРОЕКТ

Проект 1: «Интернет-магазин». Спроектировать БД из 8–10 таблиц, реализовать ограничения, заполнение тестовыми данными и набор бизнес-запросов.

Блок 2. Advanced SQL и аналитические запросы

24 ак. часа · теория + live coding + практика + code review

2.1. Подзапросы и EXISTS

- Коррелированные и некоррелированные подзапросы.
- EXISTS / NOT EXISTS, IN, ANY/ALL.
- Когда JOIN лучше подзапроса и наоборот.

2.2. CTE и рекурсивный SQL

- WITH, цепочки CTE, materialized/not materialized.
- WITH RECURSIVE: дерево категорий, оргструктура, графовые сценарии.

2.3. Оконные функции

- OVER, PARTITION BY, ORDER BY, frame clauses.
- ROW_NUMBER, RANK, DENSE_RANK, LAG, LEAD, FIRST_VALUE.
- Running total, moving average, top-N per group.

2.4. Продвинутые возможности SQL PostgreSQL

- FILTER, DISTINCT ON, LATERAL.
- GROUPING SETS, ROLLUP, CUBE.
- MERGE, keyset pagination, безопасная массовая модификация данных.

ПРАКТИЧЕСКИЙ ПРОЕКТ

Проект 2: «Аналитическая подсистема». Построить набор отчётов: продажи по периодам, рейтинг клиентов, running totals, рекурсивная структура и план-факт.

Блок 3. Проектирование и возможности PostgreSQL

24 ак. часа · теория + live coding + практика + code review

3.1. Моделирование данных

- ER-модель, сущности, связи, кардинальности.
- 1NF–3NF, BCNF. Осознанная денормализация.
- Выбор натуральных и surrogate keys. UUID и identity.

3.2. Специализированные типы PostgreSQL

- ARRAY, RANGE/MULTIRANGE, ENUM, DOMAIN, composite types.
- JSON/JSONB: структура, операторы, jsonpath.
- Когда JSONB полезен, а когда ухудшает модель.

3.3. Представления и материализация

- VIEW, security barrier концептуально.
- MATERIALIZED VIEW, REFRESH, сценарии кэширования отчётности.

3.4. Schema evolution и миграции

- Безопасные изменения схемы.
- Generated columns, expression constraints.
- Подходы к миграциям: Flyway/Liquibase концептуально.

ПРАКТИЧЕСКИЙ ПРОЕКТ

Проект 3: «CRM/SaaS Database». Спроектировать БД 15–20 таблиц с ER-диаграммой, миграциями, JSONB, ограничениями и материализованным представлением.

Блок 4. PL/pgSQL и серверная разработка

24 ак. часа · теория + live coding + практика + code review

4.1. Функции и процедуры

- SQL functions и PL/pgSQL. Параметры, типы возврата, SETOF/TABLE.
- RETURN QUERY, локальные переменные, управляющие конструкции.

4.2. Dynamic SQL и обработка ошибок

- EXECUTE, format(), безопасная динамика.
- EXCEPTION, диагностика ошибок, логирование.

4.3. Триггеры и аудит

- BEFORE/AFTER/INSTEAD OF. Row vs statement triggers.
- Transition tables, аудит изменений, контроль бизнес-правил.

4.4. Где должна жить бизнес-логика

- SQL vs PL/pgSQL vs backend-код.
- Практические кейсы Java/Spring, Python/FastAPI, PHP/Laravel.
- Advisory locks и сценарии координации.

ПРАКТИЧЕСКИЙ ПРОЕКТ

Проект 4: «Серверная логика». Реализовать функции, процедуры, триггеры аудита и бизнес-операцию с транзакционной целостностью.

Блок 5. Транзакции, MVCC и производительность

30 ак. часов · теория + live coding + практика + code review

5.1. Транзакции и изоляция

- ACID, BEGIN/COMMIT/ROLLBACK, savepoints.
- Read Committed, Repeatable Read, Serializable.
- Аномалии чтения и практическое воспроизведение конфликтов.

5.2. MVCC и внутреннее устройство

- xmin/xmax, snapshots, версии строк, dead tuples.
- HOT updates, VACUUM, autovacuum, wraparound.
- Почему UPDATE создаёт новую версию строки.

5.3. Блокировки и конкуренция

- Row/table locks, SELECT FOR UPDATE/SHARE.
- Deadlocks: диагностика и предотвращение.
- pg_locks и pg_stat_activity.

5.4. Индексы PostgreSQL

- B-tree, Hash, GIN, GiST, BRIN.
- Multicolumn, partial, expression, INCLUDE.
- Index-only scan, bitmap scan, селективность и порядок колонок.

5.5. EXPLAIN ANALYZE и планы выполнения

- Seq Scan, Index Scan, Bitmap Heap Scan.
- Nested Loop, Hash Join, Merge Join, Sort, Aggregate, Parallel Scan.
- EXPLAIN (ANALYZE, BUFFERS), чтение фактических и плановых метрик.

5.6. Статистика и оптимизатор

- ANALYZE, pg_stats, cardinality estimation.
- CREATE STATISTICS и extended statistics.
- Разбор misestimation и исправление проблем запроса/модели/индекса.

ПРАКТИЧЕСКИЙ ПРОЕКТ

Проект 5: «Performance Lab». Получить набор медленных запросов, снять планы, определить причины и добиться измеримого ускорения с обоснованием каждого изменения.

Блок 6. Production PostgreSQL, backend и highload

18 ак. часов · теория + live coding + практика + code review

6.1. Соединения и pooling

- Connection limits. Почему «по соединению на запрос» не работает.
- PgBouncer: session vs transaction pooling.
- Пулы соединений в backend-приложениях.

6.2. Партиционирование и большие таблицы

- RANGE/LIST/HASH partitioning.
- Partition pruning, выбор ключа, типичные ошибки чрезмерного партиционирования.

6.3. Мониторинг и эксплуатация

- pg_stat_database, pg_stat_user_tables, pg_stat_user_indexes.
- pg_stat_statements, long-running queries, idle in transaction.
- Backup/restore, pg_dump, основы PITR.

6.4. PostgreSQL и backend

- Prepared statements, batch operations, N+1, pagination.
- Optimistic/pessimistic locking, миграции и SQL injection.
- Практика на готовом backend-приложении.

6.5. Event-driven integration

- Проблема «транзакция + событие». Outbox Pattern.
- CDC и Debezium концептуально. PostgreSQL → Kafka.
- Когда eventual consistency оправдана.

6.6. AI-assisted SQL development

- Проверка SQL, предложенного AI.
- Поиск логических и производительных ошибок.
- EXPLAIN как обязательная верификация AI-рекомендаций.

ПРАКТИЧЕСКИЙ ПРОЕКТ

Проект 6: «Production PostgreSQL». Docker Compose: PostgreSQL + PgBouncer + приложение; настроить мониторинг, проверить нагрузочный сценарий и реализовать Outbox.

Итоговый дипломный проект — 6 ак. часов

Production-ready PostgreSQL для коммерческого приложения

Студент проектирует и защищает БД полноценного приложения. Это не «учебные 15 таблиц», а связный инженерный кейс с требованиями к данным, производительности, эксплуатации и интеграции.

- ✓ 20+ таблиц и обоснованная ER-модель;
- ✓ ограничения целостности, миграции и роли доступа;
- ✓ JSONB/специализированные типы — только там, где это оправдано;
- ✓ индексы с объяснением выбора и минимум один `partial/expression/index with INCLUDE`;
- ✓ транзакционная операция с корректным уровнем изоляции;
- ✓ PL/pgSQL-функция или процедура и аудит через `trigger`;
- ✓ не менее 5 сложных бизнес-запросов, включая окно/CTE;
- ✓ EXPLAIN ANALYZE + BUFFERS для ключевых запросов и отчёт об оптимизации;
- ✓ PgBouncer или backend pool, Docker Compose, monitoring query pack;
- ✓ Outbox Pattern или аргументированное решение без событийной интеграции.

Формат защиты

- 10–15 минут демонстрации проекта;
- ER-диаграмма и архитектурные решения;
- разбор одного сложного SQL и одного плана выполнения;
- ответы на вопросы по транзакциям, индексам и производительности.

Методика обучения

Live coding вместо «готовых слайдов»	Преподаватель пишет запросы, строит схему, воспроизводит блокировки и разбирает планы выполнения прямо на занятии.
Проект после каждого блока	Новые темы закрепляются в связном инженерном кейсе, а не в наборе несвязанных упражнений.
Code/SQL review	Студент получает обратную связь по структуре SQL, модели данных, индексам и читаемости решения.
Диагностика ошибок	В программу намеренно включаются плохие запросы, неверные индексы и конфликтующие транзакции — студент учится находить причину.
Инженерный выбор	Задача — не только знать функцию PostgreSQL, но и уметь объяснить, когда её стоит или не стоит использовать.

Технологии и инструменты

PostgreSQL	SQL / PL/pgSQL	psql
pgAdmin / DataGrip	Docker / Docker Compose	PgBouncer
pg_stat_statements	EXPLAIN ANALYZE	Git
Flyway/Liquibase (концептуально)	Java/Spring Boot (готовый backend-кейс)	Kafka / Debezium (концептуально)

Чем программа отличается от типового курса PostgreSQL

- не заканчивается на SELECT/JOIN — доходит до MVCC, планов выполнения, статистики и production-среды;
- обучает читать EXPLAIN ANALYZE, а не просто «добавлять индекс»;
- связывает PostgreSQL с backend-разработкой, pooling, Docker и событийной архитектурой;
- после каждого большого блока есть проект с code review;
- AI используется как инструмент, но каждое его предложение проверяется SQL-планом и фактическими метриками;
- финальный проект требует обосновывать инженерные решения, а не только показать работающие запросы.

Рекомендуемое позиционирование на сайте

POSTGRESQL-РАЗРАБОТЧИК: ОТ SQL ДО PRODUCTION И HIGHLOAD

Научитесь проектировать и оптимизировать PostgreSQL для реальных backend-систем: сложный SQL, PL/pgSQL, транзакции и MVCC, EXPLAIN ANALYZE, индексы, партиционирование, PgBouncer, Docker и production-практика.

150 академических часов · 6 проектов · дипломный проект

ТЕРАКОР · технологии и развитие корпоративных решений