5-Day MySQL + PostgreSQL + Sequelize Training Program

Goal

By the end of this program, trainees will:

- Understand **SQL fundamentals** (MySQL & PostgreSQL).
- Work with **Sequelize ORM** in Node.js.
- Design **real-world schemas** with normalization.
- Implement **CRUD, Joins, Relationships, Transactions, Indexes**.
- Use **advanced PostgreSQL features (JSONB, Window Functions, Full-Text Search)**.
- Learn **performance, optimization, and data security best practices**.

Day 1 — SQL & Sequelize Fundamentals

What to Learn

- RDBMS concepts (MySQL vs PostgreSQL).
- Data types (INT, VARCHAR, DATE, ENUM, JSON).
- NULL handling (`IS NULL`, `NOT NULL`).
- Primary Key vs Unique vs Foreign Key.
- Default values.
- Create DB & Tables.
- CRUD: `INSERT`, `SELECT`, `UPDATE`, `DELETE`.
- Sequelize setup + first model.

📝 Task: *Student Management System*

- Create `students` table: id, name, email, age, course.
- Insert at least 5 students.
- Update a student’s email.
- Delete a student.
- Prevent duplicate emails with `UNIQUE`.
- Repeat using Sequelize.

Deliverable: Student CRUD in MySQL/Postgres + Sequelize.

Day 2 — Relationships, Normalization & Joins

What to Learn

- Database Normalization (1NF, 2NF, 3NF).
- Foreign Keys & Cascade Delete/Update.
- Relationships: One-to-One, One-to-Many, Many-to-Many.
- Joins: INNER, LEFT, RIGHT, FULL.
- Sequelize associations (`hasOne`, `hasMany`, `belongsToMany`).

Task: *Library Management*

- Create:
  - `users`: id, name, email.
  - `books`: id, title, author, category.
  - `borrowed_books`: userId, bookId.
- Queries:
  - Books borrowed by a user.
  - Users with more than 2 borrowed books.
- Add cascade delete on user removal.
- Implement Sequelize associations.

Deliverable: Normalized Library DB with Sequelize relations.

Day 3 — Advanced Queries, Views & Transactions

What to Learn

- Filtering: `WHERE`, `LIKE`, `IN`, `BETWEEN`.
- Aggregate functions: `COUNT`, `SUM`, `AVG`, `MAX`, `MIN`.
- Grouping: `GROUP BY`, `HAVING`.
- Pagination (`LIMIT`, `OFFSET`).
- Views (virtual tables).
- Transactions & Isolation Levels.
- Sequelize Scopes + Transactions.

Task: *E-Commerce Orders*

- Create `orders`: id, userId, productId, amount, date.
- Queries:
  - Total sales per user.
  - Average order value.
  - Orders in last 7 days.
- Create a **view** for “top customers”.
- Implement order insert with Sequelize transaction.

Deliverable: E-commerce analytics queries + safe transactions.

Day 4 — Optimization, Indexing & Security

What to Learn

- Indexes: simple, composite, unique, partial (Postgres).
- Full-text search (MySQL & Postgres).
- SQL Injection prevention (parameterized queries).
- Stored Procedures & Triggers.
- Sequelize Hooks (e.g., hash password before save).
- Migrations & Seeders.
- Backup & Restore basics (`mysqldump`, `pg_dump`).
- Connection Pooling.

Task: *Banking System*

- Create `accounts`: id, userId, balance.
- Implement money transfer with Sequelize **transaction** + rollback on error.
- Add **index** on userId.
- Write a **trigger** to auto-update `updated_at` on balance change.
- Use Sequelize hook for password hashing.
- Demonstrate `mysqldump`/`pg_dump` backup.

Deliverable: Secure & optimized banking system with indexing + hooks.

Day 5 — PostgreSQL Advanced Features & Analytics

What to Learn

- PostgreSQL advanced types: Arrays, JSONB.

- Full-Text Search (`tsvector`, `tsquery`).
- Window Functions: `ROW_NUMBER`, `RANK`, `LEAD`, `LAG`.
- CTEs (WITH queries).
- Partitioning & Sharding basics.
- Schema design best practices.

Task: ***Social Media Analytics***

- Tables:
  - `users`: id, name.
  - `posts`: id, userId, content, tags (ARRAY), metadata (JSONB).
  - `likes`: id, postId, userId.

- Queries:

  - Find posts with tag = "tech".
  - Full-text search in posts.
  - Rank users by post count.
  - Find top 5 most liked posts.
  - Get previous post of each user (`LAG`).

Deliverable: Social media DB with JSONB, arrays, full-text search & ranking.

Capstone Project (End of Day 5)

Build a **Mini E-Commerce Backend**:

- User signup/login.
- CRUD for products & orders.
- MySQL: Transactions + Indexing.
- PostgreSQL: JSONB metadata, Ranking analytics.
- Sequelize ORM throughout.
