1. Database Initialization & Seeding

Database initialization refers to preparing a clean, predictable database state before running tests. Seeding means inserting predefined test data (users, products, orders, etc.) so your tests run against known and stable data. This ensures tests are repeatable and not affected by leftover data.

---

2. Setting up a Test Database

A **separate test database** is used so automated tests don't touch real production or development data.
Typical setup includes:

- Creating a dedicated test DB instance.
- Running migrations automatically before tests.
- Resetting or truncating tables before each test run.
- Seeding the database using scripts or API endpoints.

This gives clean and consistent test environments.

---

3. Using Fixtures & Custom Commands for Seeding Data

Cypress allows test data injection using:

- **Fixtures:** JSON files that store sample data used for seeding (e.g., user.json).
- **Custom Commands:** You can write commands like `cy.seedUser()` or `cy.resetDB()` that hit backend API routes or scripts to insert data.
  This avoids writing the same setup code repeatedly and keeps tests clean and readable.

---

4. Best Practices for Maintaining Test Data

- Always start tests with a **known state** (clean DB + initial seed).
- Avoid creating test data through UI steps—it slows down tests.
- Prefer **API or direct DB seeding** for speed and reliability.
- Keep fixtures updated and version-controlled.
- Use unique data identifiers to avoid conflicts during parallel runs.
- Never depend on data created by previous tests; each test should be independent.