1. Table Definitions

Below are the 10 tables with their columns, primary keys (PK), foreign keys (FK), and purposes based on the functional requirements.

user Table

* Purpose: Stores student data (FR1-FR6).
* Columns:
  + user\_id (SERIAL, PK): Unique identifier.
  + full\_name (VARCHAR(100), NOT NULL): User’s name.
  + email (VARCHAR(100), UNIQUE, NOT NULL): Login credential.
  + password (VARCHAR(255), NOT NULL): Hashed password.
  + phone\_number (VARCHAR(15)): Optional contact number.
  + profile\_picture (VARCHAR(255)): Optional image path/URL.
  + created\_at (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Registration timestamp.
* Relationships:
  + 1:N with enrollment (user\_id → enrollment.user\_id).
  + 1:N with order (user\_id → order.user\_id).
  + 1:N with password\_reset (user\_id → password\_reset.user\_id).

admin Table

* Purpose: Stores admin data (FR7-FR9).
* Columns:
  + admin\_id (SERIAL, PK): Unique identifier.
  + full\_name (VARCHAR(100), NOT NULL): Admin’s name.
  + email (VARCHAR(100), UNIQUE, NOT NULL): Login credential.
  + password (VARCHAR(255), NOT NULL): Hashed password.
  + phone\_number (VARCHAR(15)): Optional contact number.
  + created\_at (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Registration timestamp.
* Relationships:
  + 1:N with course (admin\_id → course.instructor\_id).

category Table

* Purpose: Groups courses (FR10).
* Columns:
  + category\_id (SERIAL, PK): Unique identifier.
  + name (VARCHAR(50), NOT NULL): Category name (e.g., "Web Development").
  + description (TEXT): Optional description.
* Relationships:
  + 1:N with course (category\_id → course.category\_id).

course Table

* Purpose: Stores course details (FR11, FR12).
* Columns:
  + course\_id (SERIAL, PK): Unique identifier.
  + title (VARCHAR(100), NOT NULL): Course name.
  + description (TEXT): Course details.
  + price (DECIMAL(10,2), NOT NULL, CHECK >= 0): Cost.
  + duration (VARCHAR(50)): Length (e.g., "10 hours").
  + category\_id (INT, FK): Links to category.
  + instructor\_id (INT, FK): Links to admin instructor.
  + image\_url (VARCHAR(255)): Course image path/URL.
  + created\_at (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Creation timestamp.
* Relationships:
  + N:1 with category (category\_id → category.category\_id).
  + N:1 with admin (instructor\_id → admin.admin\_id).
  + N:M with user via enrollment (course\_id → enrollment.course\_id).
  + N:1 with order\_item (course\_id → order\_item.course\_id).

enrollment Table

* Purpose: Tracks user progress in courses (FR13).
* Columns:
  + enrollment\_id (SERIAL, PK): Unique identifier.
  + user\_id (INT, FK): Links to user.
  + course\_id (INT, FK): Links to course.
  + enrollment\_date (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Enrollment timestamp.
  + progress (DECIMAL(5,2), DEFAULT 0.00, CHECK 0-100): Completion percentage.
* Relationships:
  + N:M between user and course (user\_id → user.user\_id, course\_id → course.course\_id).

order Table

* Purpose: Records purchases (FR15, FR16).
* Columns:
  + order\_id (SERIAL, PK): Unique identifier.
  + user\_id (INT, FK): Links to user.
  + total\_amount (DECIMAL(10,2), NOT NULL, CHECK >= 0): Total cost.
  + order\_date (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Purchase timestamp.
  + status (VARCHAR(20), CHECK 'pending', 'completed', 'failed', DEFAULT 'pending'): Order status.
* Relationships:
  + 1:N with user (user\_id → user.user\_id).
  + 1:N with order\_item (order\_id → order\_item.order\_id).
  + 1:1 with payment (order\_id → payment.order\_id).

order\_item Table

* Purpose: Links courses to orders (FR17).
* Columns:
  + order\_item\_id (SERIAL, PK): Unique identifier.
  + order\_id (INT, FK): Links to order.
  + course\_id (INT, FK): Links to course.
  + price (DECIMAL(10,2), NOT NULL, CHECK >= 0): Price at purchase.
* Relationships:
  + 1:N with order (order\_id → order.order\_id).
  + N:1 with course (course\_id → course.course\_id).

payment Table

* Purpose: Tracks payment details (FR18).
* Columns:
  + payment\_id (SERIAL, PK): Unique identifier.
  + order\_id (INT, UNIQUE, FK): Links to order.
  + amount (DECIMAL(10,2), NOT NULL, CHECK >= 0): Payment amount.
  + payment\_method (VARCHAR(50)): Method (e.g., "credit\_card").
  + transaction\_id (VARCHAR(100)): Transaction reference.
  + status (VARCHAR(20), CHECK 'success', 'failed'): Payment outcome.
  + payment\_date (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Payment timestamp.
* Relationships:
  + 1:1 with order (order\_id → order.order\_id).

contact\_message Table

* Purpose: Stores contact submissions (FR19).
* Columns:
  + message\_id (SERIAL, PK): Unique identifier.
  + name (VARCHAR(100), NOT NULL): Sender’s name.
  + email (VARCHAR(100), NOT NULL): Sender’s email.
  + message (TEXT, NOT NULL): Message content.
  + submitted\_at (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Submission timestamp.
* Relationships: None (standalone).

password\_reset Table

* Purpose: Manages reset tokens (FR20).
* Columns:
  + password\_reset\_id (SERIAL, PK): Unique identifier.
  + user\_id (INT, FK): Links to user.
  + token (VARCHAR(255), NOT NULL): Reset token.
  + expiration\_date (TIMESTAMP, NOT NULL, CHECK > CURRENT\_TIMESTAMP): Token expiry.
* Relationships:
  + 1:N with user (user\_id → user.user\_id).

**#Schema**

-- user Table - FR1, FR2, FR3, FR5, FR6

-- Customer Table - FR1, FR2, FR3, FR5, FR6

CREATE TABLE Customer (

user\_id SERIAL PRIMARY KEY,

full\_name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL, -- Hashed

phone\_number VARCHAR(15),

profile\_picture VARCHAR(255),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- admin Table - FR7, FR8, FR9

CREATE TABLE admin (

admin\_id SERIAL PRIMARY KEY,

full\_name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL, -- Hashed

phone\_number VARCHAR(15),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- category Table - FR10

CREATE TABLE category (

category\_id SERIAL PRIMARY KEY,

name VARCHAR(50) NOT NULL,

description TEXT

);

-- course Table - FR11, FR12

CREATE TABLE course (

course\_id SERIAL PRIMARY KEY,

title VARCHAR(100) NOT NULL,

description TEXT,

price DECIMAL(10,2) NOT NULL CHECK (price >= 0),

duration VARCHAR(50),

category\_id INT,

instructor\_id INT,

image\_url VARCHAR(255),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

CONSTRAINT fk\_category FOREIGN KEY (category\_id) REFERENCES category(category\_id) ON DELETE SET NULL,

CONSTRAINT fk\_instructor FOREIGN KEY (instructor\_id) REFERENCES admin(admin\_id) ON DELETE SET NULL

);

-- enrollment Table - FR13

CREATE TABLE enrollment (

enrollment\_id SERIAL PRIMARY KEY,

user\_id INT,

course\_id INT,

enrollment\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

progress DECIMAL(5,2) DEFAULT 0.00 CHECK (progress >= 0 AND progress <= 100),

CONSTRAINT fk\_customer\_enrollment FOREIGN KEY (user\_id) REFERENCES Customer(user\_id) ON DELETE CASCADE,

CONSTRAINT fk\_course\_enrollment FOREIGN KEY (course\_id) REFERENCES course(course\_id) ON DELETE CASCADE

);

-- order Table - FR15, FR16

CREATE TABLE "order" (

order\_id SERIAL PRIMARY KEY,

user\_id INT,

total\_amount DECIMAL(10,2) NOT NULL CHECK (total\_amount >= 0),

order\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status VARCHAR(20) CHECK (status IN ('pending', 'completed', 'failed')) DEFAULT 'pending',

CONSTRAINT fk\_customer\_order FOREIGN KEY (user\_id) REFERENCES Customer(user\_id) ON DELETE CASCADE

);

-- order\_item Table - FR17

CREATE TABLE order\_item (

order\_item\_id SERIAL PRIMARY KEY,

order\_id INT,

course\_id INT,

price DECIMAL(10,2) NOT NULL CHECK (price >= 0),

CONSTRAINT fk\_order\_item FOREIGN KEY (order\_id) REFERENCES "order"(order\_id) ON DELETE CASCADE,

CONSTRAINT fk\_course\_item FOREIGN KEY (course\_id) REFERENCES course(course\_id) ON DELETE SET NULL

);

-- payment Table - FR18

CREATE TABLE payment (

payment\_id SERIAL PRIMARY KEY,

order\_id INT UNIQUE,

amount DECIMAL(10,2) NOT NULL CHECK (amount >= 0),

payment\_method VARCHAR(50),

transaction\_id VARCHAR(100),

status VARCHAR(20) CHECK (status IN ('success', 'failed')),

payment\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

CONSTRAINT fk\_payment\_order FOREIGN KEY (order\_id) REFERENCES "order"(order\_id) ON DELETE SET NULL

);

-- contact\_message Table - FR19

CREATE TABLE contact\_message (

message\_id SERIAL PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) NOT NULL,

message TEXT NOT NULL,

submitted\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- password\_reset Table - FR20

CREATE TABLE password\_reset (

password\_reset\_id SERIAL PRIMARY KEY,

user\_id INT,

token VARCHAR(255) NOT NULL,

expiration\_date TIMESTAMP NOT NULL CHECK (expiration\_date > CURRENT\_TIMESTAMP),

CONSTRAINT fk\_customer\_reset FOREIGN KEY (user\_id) REFERENCES Customer(user\_id) ON DELETE CASCADE

);

