```
PRAGMA foreign_keys = ON;

-- ============= ROLES & USERS (Auth Layer) =============

CREATE TABLE roles (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL UNIQUE      -- student | doctor | assistant | admin
);

CREATE TABLE users (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  username TEXT NOT NULL UNIQUE,
  email TEXT NOT NULL UNIQUE,
  password_hash TEXT NOT NULL,
  role_id INTEGER NOT NULL,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (role_id) REFERENCES roles(id)
);

-- ============= CORE DOMAIN TABLES =============

CREATE TABLE Department (
  department_id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL UNIQUE,
  description TEXT,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
```

```
);

CREATE TABLE Student (
  student_id INTEGER PRIMARY KEY AUTOINCREMENT,
  student_number TEXT NOT NULL UNIQUE,
  full_name TEXT NOT NULL,
  email TEXT NOT NULL UNIQUE,
  phone TEXT,
  year_level TEXT NOT NULL CHECK (year_level IN ('Fourth','Fifth','Internship')),
  status TEXT NOT NULL DEFAULT 'Active' CHECK (status IN ('Active','OnLeave','Graduated','Dismissed')),
  graduation_year INTEGER,
  current_status TEXT,
  notes TEXT,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  user_id INTEGER UNIQUE,
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);

CREATE TABLE Instructor (
  instructor_id INTEGER PRIMARY KEY AUTOINCREMENT,
  full_name TEXT NOT NULL,
  email TEXT UNIQUE,
  role TEXT NOT NULL CHECK (role IN ('Professor','AssistantProfessor','Lecturer','TA')),
  department_id INTEGER,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  user_id INTEGER UNIQUE,
  FOREIGN KEY (department_id) REFERENCES Department(department_id),
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);

CREATE TABLE AssignmentType (
  type_id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL UNIQUE,
  description TEXT,
  allowed_file_types TEXT NOT NULL CHECK (allowed_file_types IN
('Image','PPT','PDF','Other','Multiple')),
  is_active INTEGER NOT NULL DEFAULT 1 CHECK (is_active IN (0,1))
);

CREATE TABLE Assignment (
  assignment_id INTEGER PRIMARY KEY AUTOINCREMENT,
  title TEXT NOT NULL,
  description TEXT,
  type_id INTEGER NOT NULL,
  department_id INTEGER NOT NULL,
  created_by INTEGER NOT NULL,
  target_year TEXT NOT NULL CHECK (target_year IN ('Fourth','Fifth','Internship','All')),
  deadline DATETIME NOT NULL,
  max_grade REAL DEFAULT 100.00,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (type_id) REFERENCES AssignmentType(type_id) ON UPDATE CASCADE,
  FOREIGN KEY (department_id) REFERENCES Department(department_id) ON UPDATE
CASCADE,
  FOREIGN KEY (created_by) REFERENCES Instructor(instructor_id) ON UPDATE CASCADE
);

CREATE TABLE Submission (
  submission_id INTEGER PRIMARY KEY AUTOINCREMENT,
  assignment_id INTEGER NOT NULL,
  student_id INTEGER NOT NULL,
  original_filename TEXT NOT NULL,
  file_path TEXT NOT NULL,
  file_type TEXT NOT NULL CHECK (file_type IN ('Image','PPT','PDF','Other')),
  submitted_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  status TEXT NOT NULL DEFAULT 'Pending' CHECK (status IN
('Pending','Accepted','Rejected','NeedsRevision')),
  student_notes TEXT,
  FOREIGN KEY (assignment_id) REFERENCES Assignment(assignment_id) ON UPDATE
CASCADE ON DELETE CASCADE,
  FOREIGN KEY (student_id) REFERENCES Student(student_id) ON UPDATE CASCADE ON
DELETE CASCADE,
  UNIQUE (assignment_id, student_id)
);

CREATE TABLE SubmissionFeedback (
  feedback_id INTEGER PRIMARY KEY AUTOINCREMENT,
  submission_id INTEGER NOT NULL UNIQUE,
  instructor_id INTEGER NOT NULL,
  feedback_text TEXT,
  grade REAL,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (submission_id) REFERENCES Submission(submission_id) ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY (instructor_id) REFERENCES Instructor(instructor_id) ON UPDATE CASCADE
);

-- Helpful indexes
CREATE INDEX idx_assignment_deadline ON Assignment(deadline);
CREATE INDEX idx_submission_student ON Submission(student_id);
CREATE INDEX idx_submission_assignment ON Submission(assignment_id);
CREATE INDEX idx_feedback_grade ON SubmissionFeedback(grade);
```