# CDAC Mumbai

## Module: WPT

## Database CRD Operations with Express.js & MySQL

- Create a Database -> `practice_db`
- install `mysql2` package: `npm install mysql2`
- install `express` package: `npm install express`

## Implement the Following APIs

**Create a new table:**

```
CREATE TABLE students (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    age INT,
    course VARCHAR(10),
    email VARCHAR(100)
);
```

### Q1. Database Connection Setup

- Create a MySQL connection using with credentials

### Q2. Welcome Route

Create a GET route `/` that returns:

```
{ "message": "Welcome to Practice API" }
```

**HTTP Status**: 200

### Q3. Get All Students

Create route: `GET /students`

**Requirements:**

- Query: `SELECT * FROM students`
- Return status 200 with data array
- Handle errors with status 500 and message: `{ "message": "Failed to fetch students" }`

## Q4. Get Student by ID

Create route: `GET /students/:id`

**Example Request:** `/students/1`

**Requirements:**

- Use URL parameter `:id`
- Query: `SELECT * FROM students WHERE id = ?`
- If student exists: return status 200 with student object
- If not found: return status 404 with error message
- Handle database errors with status 500

---

## Q5. Delete Student by ID

Create route: `DELETE /students/:id`

**Example Request:** `DELETE /students/3`

**Requirements:**

- Use URL parameter `:id`
- Query: `DELETE FROM students WHERE id = ?`
- Check `affectedRows` from result
- If deleted (affectedRows > 0): return status 200
- If not found (affectedRows === 0): return status 404
- Handle errors with status 500

---

## Q6. Get Students by Course

Create route: `GET /students/course/:course`

**Example Request:** `/students/course/PGDAC`

**Response:**

```
[
  { "id": 1, "name": "Tony Stark", "age": 16, "course": "PGDAC" },
  { "id": 4, "name": "Steve Rogers", "age": 16, "course": "PGDAC" }
]
```

**Requirements:**

- Use URL parameter `:course`
- Query: `SELECT * FROM students WHERE course = ?`
- Return empty array if no students found
- Status 200 for success, 500 for errors

---

## Q7. Create Library Management API

**Create a new table:**

```
CREATE TABLE books (
    id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(200) NOT NULL,
    author VARCHAR(100),
    price DECIMAL(10, 2),
    available BOOLEAN DEFAULT true
);
```

**Implement these routes:**

1. `GET /books` - Get all books
2. `GET /books/:id` - Get book by ID
3. `DELETE /books/:id` - Delete a book
4. `GET /books/author/:authorName` - Get all books by specific author
5. `GET /books/available` - Get only available books (where available = true)

**Example Response for available books:**

```
[
  { "id": 1, "title": "The Alchemist", "author": "Paulo Coelho", "price": 350,
"available": true }
]
```

## Q8. Create Product Inventory API

**Create table:**

```
CREATE TABLE products (
    id INT PRIMARY KEY AUTO_INCREMENT,
    product_name VARCHAR(150),
    category VARCHAR(50),
    quantity INT,
    price DECIMAL(10, 2)
);
```

**Implement:**

1. `GET /products` - All products
2. `GET /products/:id` - Product by ID
3. `DELETE /products/:id` - Delete product

4. `GET /products/category/:category` - Products by category
5. `GET /products/low-stock` - Products where quantity < 10

---

*Remember: Practice makes perfect. Don't just copy-paste, understand each line of code!*