```swift
//
//  ViewController.swift
//  sqlite
//
//  Created by ICT2Batch1 on 27/02/26.
//
import UIKit
import SQLite3
struct Employee {
    var id: Int
    var name: String
    var age: Int
}
class ViewController: UIViewController, UITableViewDelegate, UITableViewDataSource
{


    @IBOutlet weak var rollInput: UITextField!
    @IBOutlet weak var nameInput: UITextField!
    @IBOutlet weak var ageInput: UITextField!
    @IBOutlet weak var tableView: UITableView!

    let dbPath = "emp.sqlite"
        var db: OpaquePointer?
        var empList = [Employee]()


    override func viewDidLoad() {
        super.viewDidLoad()
        tableView.delegate = self
                tableView.dataSource = self

                db = openDatabase()
                createTable()
                loadData()
    }

        func loadData() {
            empList = read()
            tableView.reloadData()
    }



func tableView(_ tableView: UITableView,
                   numberOfRowsInSection section: Int) -> Int {
        return empList.count
    }

    func tableView(_ tableView: UITableView,
                   cellForRowAt indexPath: IndexPath) -> UITableViewCell {

        let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)
        let emp = empList[indexPath.row]

        cell.textLabel?.text = emp.name + "ID: \(emp.id) | Age: \(emp.age)"
        cell.detailTextLabel?.text = "ID: \(emp.id) | Age: \(emp.age)"

        return cell
    }

func tableView(_ tableView: UITableView,
                  didSelectRowAt indexPath: IndexPath) {

       let emp = empList[indexPath.row]
       rollInput.text = "\(emp.id)"
       nameInput.text = emp.name
       ageInput.text = "\(emp.age)"
   }

   // MARK: - Button Actions


@IBAction func insertAction(_ sender: UIButton) {
    insert(id: Int(rollInput.text!) ?? 0,
                  name: nameInput.text ?? "",
                  age: Int(ageInput.text!) ?? 0)
           clearFields()
           loadData()
}
```

```
@IBAction func updateAction(_ sender: UIButton) {

    update(id: Int(rollInput.text!) ?? 0,
                  name: nameInput.text ?? "",
                  age: Int(ageInput.text!) ?? 0)
           clearFields()
           loadData()
}


@IBAction func deleteAction(_ sender: UIButton) {

    deleteByID(id: Int(rollInput.text!) ?? 0)
           clearFields()
           loadData()
}

func clearFields() {
        rollInput.text = ""
        nameInput.text = ""
        ageInput.text = ""
    }

    // MARK: - Open Database
    func openDatabase() -> OpaquePointer? {

        let fileURL = try! FileManager.default
            .url(for: .documentDirectory,
                 in: .userDomainMask,
                 appropriateFor: nil,
                 create: false)
            .appendingPathComponent(dbPath)

        var db: OpaquePointer?

        if sqlite3_open(fileURL.path, &db) != SQLITE_OK {
            print("Unable to open database")
            return nil
        }

        print("Database opened successfully", fileURL.path)
        return db
    }

    // MARK: - Create Table
    func createTable() {

        let query = """
        CREATE TABLE IF NOT EXISTS person(
        Id INTEGER PRIMARY KEY,
        name TEXT,
        age INTEGER);
        """

        var statement: OpaquePointer?

        if sqlite3_prepare_v2(db, query, -1, &statement, nil) == SQLITE_OK {
            if sqlite3_step(statement) == SQLITE_DONE {
                print("Table created")
            }
        }

        sqlite3_finalize(statement)
    }

    // MARK: - Insert
    func insert(id: Int, name: String, age: Int) {

        let query = "INSERT INTO person (Id, name, age) VALUES (?, ?, ?);"
        var statement: OpaquePointer?

        if sqlite3_prepare_v2(db, query, -1, &statement, nil) == SQLITE_OK {

            sqlite3_bind_int(statement, 1, Int32(id))
            sqlite3_bind_text(statement, 2, (name as NSString).utf8String, -1, nil)
            sqlite3_bind_int(statement, 3, Int32(age))

            sqlite3_step(statement)
        }

        sqlite3_finalize(statement)
    }

    // MARK: - Read
    func read() -> [Employee] {

        let query = "SELECT * FROM person;"
        var statement: OpaquePointer?
        var list = [Employee]()

        if sqlite3_prepare_v2(db, query, -1, &statement, nil) == SQLITE_OK {

            while sqlite3_step(statement) == SQLITE_ROW {

                let id = sqlite3_column_int(statement, 0)
                let name = String(cString: sqlite3_column_text(statement, 1))
                let age = sqlite3_column_int(statement, 2)

                list.append(Employee(id: Int(id),
                                     name: name,
                                     age: Int(age)))
            }
        }

        sqlite3_finalize(statement)
        return list
    }

    // MARK: - Update
    func update(id: Int, name: String, age: Int) {

        let query = "UPDATE person SET name = ?, age = ? WHERE Id = ?;"
        var statement: OpaquePointer?

        if sqlite3_prepare_v2(db, query, -1, &statement, nil) == SQLITE_OK {

            sqlite3_bind_text(statement, 1, (name as NSString).utf8String, -1, nil)
            sqlite3_bind_int(statement, 2, Int32(age))
```

```
            sqlite3_bind_int(statement, 3, Int32(id))

            sqlite3_step(statement)
        }

        sqlite3_finalize(statement)
    }

    // MARK: - Delete
    func deleteByID(id: Int) {

        let query = "DELETE FROM person WHERE Id = ?;"
        var statement: OpaquePointer?

        if sqlite3_prepare_v2(db, query, -1, &statement, nil) == SQLITE_OK {

            sqlite3_bind_int(statement, 1, Int32(id))
            sqlite3_step(statement)
        }

        sqlite3_finalize(statement)
    }

}
```