Module : 04

SQLite3 Exercise

Name : Anubhav Ranjan ***Registration No : 774***

College : IIIT kalyani ***Roll No : ECE/21114***

Scenario: Employee Management System

Background:

A company maintains a database with two tables:

- **Employees: Stores information about employees.**

EmployeeID	Name	DepartmentID	Salary	HireDate
1	Alice	101	70000	2021-01-15
2	Bob	102	60000	2020-03-10
3	Charlie	101	80000	2022-05-20
4	Diana	103	75000	2019-07-25

- **Departments: Stores information about departments.**

DepartmentID	DepartmentName
101	HR
102	IT
103	Finance

SetUp: Database and Table Creation

Open the command prompt/terminal to start the SQLite3 Command-Line Interface (CLI) or use a database management tool like DB Browser for SQLite:

-- To create or Use Database:

-> sqlite3 EmployeeManagement.db

```
PS C:\Users\Anubhav Ranjan\OneDrive\Desktop\NexTurn\NexTurn\AnubhavRanjan-NexTurn-Program\M4_Data_Engineering_Assignment
s\cs1_Employee_Management_System_SQLite3> sqlite3 EmployeeManagement.db
SQLite version 3.46.1 2024-08-13 09:16:08 (UTF-16 console I/O)
Enter ".help" for usage hints.
sqlite>
```

-- To Create the `Departments` Table:

-> CREATE TABLE Departments (
DepartmentID INTEGER PRIMARY KEY,
DepartmentName TEXT NOT NULL
);

```
sqlite> CREATE TABLE Departments (
(x1...>     DepartmentID INTEGER PRIMARY KEY,
(x1...>     DepartmentName TEXT NOT NULL
(x1...> );
sqlite> .tables
Departments
sqlite> .schema Departments
CREATE TABLE Departments (
    DepartmentID INTEGER PRIMARY KEY,
    DepartmentName TEXT NOT NULL
);
```

-- To Create the `Employees` Table:

-> CREATE TABLE Employees (
EmployeeID INTEGER PRIMARY KEY,
Name TEXT NOT NULL,
DepartmentID INTEGER NOT NULL,
Salary REAL NOT NULL,
HireDate TEXT NOT NULL,
FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);

```
sqlite> CREATE TABLE Employees (
(x1...>     EmployeeID INTEGER PRIMARY KEY,
(x1...>     Name TEXT NOT NULL,
(x1...>     DepartmentID INTEGER NOT NULL,
(x1...>     Salary REAL NOT NULL,
(x1...>     HireDate TEXT NOT NULL,
(x1...>     FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
(x1...> );
sqlite> .tables
Departments  Employees
sqlite> .schema Employees
CREATE TABLE Employees (
    EmployeeID INTEGER PRIMARY KEY,
    Name TEXT NOT NULL,
    DepartmentID INTEGER NOT NULL,
    Salary REAL NOT NULL,
    HireDate TEXT NOT NULL,
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);
```

SetUp: Inserting the Given Sample Data

-- To Insert Data into `Departments` Table:

-> INSERT INTO Departments (DepartmentID, DepartmentName) VALUES

(101, 'HR'),

(102, 'IT'),

(103, 'Finance');

```
sqlite> INSERT INTO Departments (DepartmentID, DepartmentName) VALUES
   ...> (101, 'HR'),
   ...> (102, 'IT'),
   ...> (103, 'Finance');
sqlite> SELECT * FROM Departments;
101|HR
102|IT
103|Finance
```

-- To Insert Data into `Employees` Table:

-> INSERT INTO Employees (EmployeeID, Name, DepartmentID, Salary, HireDate) VALUES

(1, 'Alice', 101, 70000, '2021-01-15'),

(2, 'Bob', 102, 60000, '2020-03-10'),

(3, 'Charlie', 101, 80000, '2022-05-20'),

(4, 'Diana', 103, 75000, '2019-07-25');

```
sqlite> INSERT INTO Employees (EmployeeID, Name, DepartmentID, Salary, HireDate) VALUES
   ...> (1, 'Alice', 101, 70000, '2021-01-15'),
   ...> (2, 'Bob', 102, 60000, '2020-03-10'),
   ...> (3, 'Charlie', 101, 80000, '2022-05-20'),
   ...> (4, 'Diana', 103, 75000, '2019-07-25');
sqlite> SELECT * FROM Employees;
1|Alice|101|70000.0|2021-01-15
2|Bob|102|60000.0|2020-03-10
3|Charlie|101|80000.0|2022-05-20
4|Diana|103|75000.0|2019-07-25
```

SQLite3 Queries:

-- Query 1: List the names of employees hired after January 1, 2021.

-> SELECT Name

FROM Employees

WHERE HireDate > '2021-01-01';

```
sqlite> SELECT Name
   ...> FROM Employees
   ...> WHERE HireDate > '2021-01-01';
Alice
Charlie
```

-- Query 2: Calculate the average salary of employees in each department.

-> SELECT D.DepartmentName, AVG(E.Salary) AS AverageSalary

FROM Employees E

JOIN Departments D ON E.DepartmentID = D.DepartmentID

GROUP BY D.DepartmentName;

```
sqlite> SELECT D.DepartmentName, AVG(E.Salary) AS AverageSalary
   ...> FROM Employees E
   ...> JOIN Departments D ON E.DepartmentID = D.DepartmentID
   ...> GROUP BY D.DepartmentName;
Finance|75000.0
HR|75000.0
IT|60000.0
```

-- Query 3: Find the department name where the total salary is the highest.

-> SELECT D.DepartmentName

FROM Employees E

JOIN Departments D ON E.DepartmentID = D.DepartmentID

GROUP BY D.DepartmentName

ORDER BY SUM(E.Salary) DESC

LIMIT 1;

```
sqlite> SELECT D.DepartmentName
   ...> FROM Employees E
   ...> JOIN Departments D ON E.DepartmentID = D.DepartmentID
   ...> GROUP BY D.DepartmentName
   ...> ORDER BY SUM(E.Salary) DESC
   ...> LIMIT 1;
HR
```

-- Query 4: List all departments that currently have no employees assigned.

-> SELECT D.DepartmentName

FROM Departments D

WHERE D.DepartmentID NOT IN (SELECT DISTINCT E.DepartmentID FROM Employees E);

```
sqlite> INSERT INTO Departments (DepartmentID, DepartmentName) VALUES(104, 'Engineering');
sqlite> SELECT * FROM Departments;
101|HR
102|IT
103|Finance
104|Engineering
sqlite> SELECT D.DepartmentName
   ...> FROM Departments D
   ...> WHERE D.DepartmentID NOT IN (SELECT DISTINCT E.DepartmentID FROM Employees E);
Engineering
```

-- Query 5: Fetch all employee details along with their department names.

-> SELECT E.*, D.DepartmentName

FROM Employees E

JOIN Departments D ON E.DepartmentID = D.DepartmentID;

```
sqlite> SELECT E.*, D.DepartmentName
   ...> FROM Employees E
   ...> JOIN Departments D ON E.DepartmentID = D.DepartmentID;
1|Alice|101|70000.0|2021-01-15|HR
2|Bob|102|60000.0|2020-03-10|IT
3|Charlie|101|80000.0|2022-05-20|HR
4|Diana|103|75000.0|2019-07-25|Finance
```