In [2]:
```
import sqlite3

# Connect to SQLite database
conn = sqlite3.connect('company.db', timeout=10)
conn.execute("PRAGMA foreign_keys = ON;")
cursor = conn.cursor()
```

In [3]:
```
# Drop existing tables to avoid conflicts
cursor.execute("DROP TABLE IF EXISTS Leave_Record;")
cursor.execute("DROP TABLE IF EXISTS Employee;")
cursor.execute("DROP TABLE IF EXISTS Department;")

# Create Department table
cursor.execute("""
    CREATE TABLE Department (
        Dept_Code INTEGER PRIMARY KEY,
        Dept_Name TEXT NOT NULL
    );
""")

# Create Employee table
cursor.execute("""
    CREATE TABLE Employee (
        Emp_Code INTEGER PRIMARY KEY,
        Emp_Name TEXT NOT NULL CHECK (Emp_Name = UPPER(Emp_Name)),
        Address TEXT,
        City TEXT,
        Basic_Salary REAL NOT NULL CHECK (Basic_Salary BETWEEN 50000 AND 90000),
        Date_of_Join DATE DEFAULT CURRENT_DATE,
        Grade TEXT CHECK (Grade IN ('A', 'B', 'C')),
        Dept_Code INTEGER,
        CONSTRAINT fk_dept FOREIGN KEY (Dept_Code)
            REFERENCES Department(Dept_Code)
            ON DELETE RESTRICT
    );
""")

# Create Leave_Record table
cursor.execute("""
    CREATE TABLE Leave_Record (
        Leave_ID INTEGER PRIMARY KEY,
        Emp_Code INTEGER,
        Leave_Type TEXT CHECK (Leave_Type IN ('CL', 'EL', 'ML')),
        From_Date DATE NOT NULL,
        To_Date DATE NOT NULL,
        CONSTRAINT fk_emp FOREIGN KEY (Emp_Code)
            REFERENCES Employee(Emp_Code)
            ON DELETE CASCADE
    );
""")

conn.commit()
```

In [4]:
```
# Insert data into Department table
cursor.execute("INSERT INTO Department VALUES (1, 'IT'), (2, 'HR'), (3, 'Finance');")

# Insert data into Employee table
cursor.execute("""
    INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, Dept_Code)
    VALUES
    (101, 'SWAPNAMOY', '123 Salt Lake', 'Kolkata', 60000, 'A', 1),
    (102, 'KRISTIDHAR', '456 Garia', 'Kolkata', 70000, 'B', 2),
    (103, 'SHAHIR', '789 Jadavpur', 'Kolkata', 75000, 'C', 3),
    (104, 'ABHI', '123 Ballygunge', 'Kolkata', 80000, 'A', 1);
""")

# Insert data into Leave_Record table
cursor.execute("""
    INSERT INTO Leave_Record (Leave_ID, Emp_Code, Leave_Type, From_Date, To_Date)
    VALUES
    (1, 101, 'CL', '2025-03-20', '2025-03-22'),
    (2, 102, 'EL', '2025-04-10', '2025-04-12'),
    (3, 103, 'ML', '2025-05-01', '2025-05-05');
```

```
""")

conn.commit()
```

In [5]:
```
print("Before deleting Employee with Emp_Code = 101:")
for row in cursor.execute("SELECT * FROM Leave_Record"):
    print(row)

# Delete employee (should delete associated leave records)
cursor.execute("DELETE FROM Employee WHERE Emp_Code = 101;")

print("\nAfter deleting Employee with Emp_Code = 101:")
for row in cursor.execute("SELECT * FROM Leave_Record"):
    print(row)

conn.commit()
```

```
Before deleting Employee with Emp_Code = 101:
(1, 101, 'CL', '2025-03-20', '2025-03-22')
(2, 102, 'EL', '2025-04-10', '2025-04-12')
(3, 103, 'ML', '2025-05-01', '2025-05-05')

After deleting Employee with Emp_Code = 101:
(2, 102, 'EL', '2025-04-10', '2025-04-12')
(3, 103, 'ML', '2025-05-01', '2025-05-05')
```

In [7]:
```
cursor.execute("DELETE FROM Department WHERE Dept_Code = 1;")
```

```
---------------------------------------------------
IntegrityError      Traceback (most recent call last)
Cell In[7], line 1
----> 1 cursor.execute("DELETE FROM Department WHERE Dept_Code = 1;")

IntegrityError: FOREIGN KEY constraint failed
```

In [8]:
```
cursor.execute("""INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, Dept_Cod
VALUES (201, 'Swapnamoy', '123 Salt Lake', 'Kolkata', 60000, 'A', 1);
""")
```

```
---------------------------------------------------
IntegrityError      Traceback (most recent call last)
Cell In[8], line 1
----> 1 cursor.execute("""INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, D
ept_Code)
      2 VALUES (201, 'Swapnamoy', '123 Salt Lake', 'Kolkata', 60000, 'A', 1);
      3 """)

IntegrityError: CHECK constraint failed: Emp_Name = UPPER(Emp_Name)
```

In [9]:
```
cursor.execute("""INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, Dept_Cod
VALUES (202, 'KRISTIDHAR', '456 Garia', 'Kolkata', 40000, 'B', 1);""")
```

```
---------------------------------------------------
IntegrityError      Traceback (most recent call last)
Cell In[9], line 1
----> 1 cursor.execute("""INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, D
ept_Code)
      2 VALUES (202, 'KRISTIDHAR', '456 Garia', 'Kolkata', 40000, 'B', 1);""")

IntegrityError: CHECK constraint failed: Basic_Salary BETWEEN 50000 AND 90000
```

In [10]:
```
cursor.execute("""INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, Dept_Cod
VALUES (203, 'SHAHIR', '789 Jadavpur', 'Kolkata', 70000, 'D', 1);""")
```

```
---------------------------------------------------
IntegrityError      Traceback (most recent call last)
Cell In[10], line 1
----> 1 cursor.execute("""INSERT INTO Employee (Emp_Code, Emp_Name, Address, City, Basic_Salary, Grade, D
ept_Code)
      2 VALUES (203, 'SHAHIR', '789 Jadavpur', 'Kolkata', 70000, 'D', 1);""")

IntegrityError: CHECK constraint failed: Grade IN ('A', 'B', 'C')
```

In [12]:
```
cursor.execute("DROP TABLE IF EXISTS Emp_Filtered;")

cursor.execute("""
```

```
    CREATE TABLE Emp_Filtered AS
    SELECT E.Emp_Code, E.Emp_Name, D.Dept_Name, E.Basic_Salary
    FROM Employee E
    JOIN Department D ON E.Dept_Code = D.Dept_Code
    WHERE D.Dept_Name = 'HR' AND E.Basic_Salary = 70000;""")
print("\n--- Table Created with Filtered Data ---")
for row in cursor.execute("SELECT * FROM Emp_Filtered"):
    print(row)
conn.commit()
```

```
--- Table Created with Filtered Data ---
(102, 'KRISTIDHAR', 'HR', 70000.0)
```

In [13]:
```
cursor.execute("""
    INSERT INTO Emp_Filtered (Emp_Code, Emp_Name, Dept_Name, Basic_Salary)
    SELECT E.Emp_Code, E.Emp_Name, D.Dept_Name, E.Basic_Salary
    FROM Employee E
    JOIN Department D ON E.Dept_Code = D.Dept_Code
    WHERE E.Basic_Salary >= 70000;
""")

print("\n--- After Inserting Employees with Basic Salary >= 70000 ---")
for row in cursor.execute("SELECT * FROM Emp_Filtered"):
    print(row)
```

```
--- After Inserting Employees with Basic Salary >= 70000 ---
(102, 'KRISTIDHAR', 'HR', 70000.0)
(102, 'KRISTIDHAR', 'HR', 70000.0)
(103, 'SHAHIR', 'Finance', 75000.0)
(104, 'ABHI', 'IT', 80000.0)
```

In [14]:
```
# Step 2c: Alter the table to add a new column 'Net_Pay'
cursor.execute("ALTER TABLE Emp_Filtered ADD COLUMN Net_Pay REAL;")

# Step 2d: Update the 'Net_Pay' with 1.5 * Basic Salary
cursor.execute("""
    UPDATE Emp_Filtered
    SET Net_Pay = 1.5 * Basic_Salary;
""")
```

Out[14]:
```
<sqlite3.Cursor at 0x249e22b11c0>
```

In [15]:
```
print("\n--- After Updating Net Pay ---")
for row in cursor.execute("SELECT * FROM Emp_Filtered"):
    print(row)
```

```
--- After Updating Net Pay ---
(102, 'KRISTIDHAR', 'HR', 70000.0, 105000.0)
(102, 'KRISTIDHAR', 'HR', 70000.0, 105000.0)
(103, 'SHAHIR', 'Finance', 75000.0, 112500.0)
(104, 'ABHI', 'IT', 80000.0, 120000.0)
```

In [17]:
```
cursor.execute("ALTER TABLE Emp_Filtered DROP COLUMN Net_Pay;")
```

```
---------------------------------------------------
ProgrammingError    Traceback (most recent call last)
Cell In[17], line 1
----> 1 cursor.execute("ALTER TABLE Emp_Filtered DROP COLUMN Net_Pay;")
      2 for row in cursor.execute("SELECT * FROM Emp_Filtered"):
      3     print(row)

ProgrammingError: Cannot operate on a closed cursor.
```

In [ ]:
```

```