```
import sqlite3
# connect to SQLite database
conn = sqlite3.connect('memory:')
cursor = conn.cursor()

print("establish in memory database connection")

# create users table
cursor.execute('''CREATE TABLE IF NOT EXISTS users (
                    id INTEGER PRIMARY KEY,
                    name TEXT,
                    balance REAL
                    )''')
# add/insert date
cursor.execute("INSERT INTO users (name, balance) VALUES (?, ?)", ('Alice', 1000.0))
cursor.execute("INSERT INTO users (name, balance) VALUES (?, ?)", ('Bob', 500.0))
# funtion to handle transfer funds transaction
def transfer_funds(sender, recipient, amount):
  try:
    # check if transaction is active
    if not conn.in_transaction:
      # start transaction
      conn.execute("BEGIN")

      # check if sender has suffecient balance
      cursor.execute("SELECT balance FROM users WHERE name=?", (sender,))
      sender_balance=cursor.fetchone()[0]
      if sender_balance < amount:
        raise ValueError("Insufficient funds")

        # update sender's balance
        cursor.execute("UPDATE users SET balance = balance - ? WHERE name=?", (amount, sender))
        # update reciept's balance
        cursor.exectue("UPDATE users SET balance = balance + ? WHERE name=?", (amount, reciept))

        #commit transaction
        if not conn.in_transaction:
            # commit only if not already in a transaction
            conn.commit()
      print("Transaction successful")
  except Exception as e:
      # rollback only if not already in a transaction if any error occurs
      if not conn.in_transaction:
          # rolback only if not already in a transaction
          conn.rollback()
      print(f"Transaction failed:  {e}")

print("created function to handle transfer of funds")
```

```
establish in memory database connection
created function to handle transfer of funds
```

```
import sqlite3
```

```
# connect to SQLite database
conn = sqlite3.connect('memory:')
cursor = conn.cursor()

print("establish in memory database connection")
```

```
establish in memory database connection
```

```
# create users table
cursor.execute('''CREATE TABLE IF NOT EXISTS users (
                  id INTEGER PRIMARY KEY,
                  name TEXT,
                  balance REAL
                  )''')
```

```
<sqlite3.Cursor at 0x79e48e270640>
```

```
# add/insert date
cursor.execute("INSERT INTO users (name, balance) VALUES (?, ?)", ('Alice', 1000.0))
cursor.execute("INSERT INTO users (name, balance) VALUES (?, ?)", ('Bob', 500.0))
```

```
<sqlite3.Cursor at 0x79e48e270640>
```

```
# funtion to handle transfer funds transaction
def transfer_funds(sender, recipient, amount):
  try:
    # check if transaction is active
    if not conn.in_transaction:
      # start transaction
      conn.execute("BEGIN")

      # check if sender has suffecient balance
      cursor.execute("SELECT balance FROM users WHERE name=?", (sender,))
      sender_balance=cursor.fetchone()[0]
      if sender_balance < amount:
        raise ValueError("Insufficient funds")

        # update sender's balance
        cursor.execute("UPDATE users SET balance = balance - ? WHERE name=?", (amount, sender))
        # update reciept's balance
        cursor.exectue("UPDATE users SET balance = balance + ? WHERE name=?", (amount, reciept))

        #commit transaction
        if not conn.in_transaction:
            # commit only if not already in a transaction
            conn.commit()
      print("Transaction successful")
  except Exception as e:
      # rollback only if not already in a transaction if any error occurs
      if not conn.in_transaction:
          # rolback only if not already in a transaction
          conn.rollback()
      print(f"Transaction failed:  {e}")

print("created function to handle transfer of funds")
```

```
created function to handle transfer of funds
```

```
# perform a fund transfer
transfer_funds('Alice', 'Bob', 200.0)
```

```
# display balances after tranasction
cursor.execute("SELECT name, balance FROM users")
print(cursor.fetchall())
```

```
[('Alice', 1000.0), ('Bob', 500.0)]
```

```
# close database connection
conn.close()

print("close database connecction")
```

```
close database connecction
```