**PL/SQL Exercise 1: Control Structures**

**CREATE TABLES AND INSERT SAMPLE DATA**

// Create CUSTOMERS table  
CREATE TABLE customers (  
customer\_id NUMBER PRIMARY KEY,  
name VARCHAR2(100),  
age NUMBER,  
balance NUMBER(10, 2),  
isvip VARCHAR2(5)  
);  
  
// Create LOANS table  
CREATE TABLE loans (  
loan\_id NUMBER PRIMARY KEY,  
customer\_id NUMBER,  
interest\_rate NUMBER(5, 2),  
due\_date DATE,  
FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)  
);  
  
// Create SAVINGS\_ACCOUNTS table  
CREATE TABLE savings\_accounts (  
account\_id NUMBER PRIMARY KEY,  
customer\_id NUMBER,  
balance NUMBER(10, 2),  
FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)  
);  
  
// Create ACCOUNTS table (for transfers)  
CREATE TABLE accounts (  
account\_id NUMBER PRIMARY KEY,  
customer\_id NUMBER,  
balance NUMBER(10, 2),  
FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)  
);

// Create EMPLOYEES table  
CREATE TABLE employees (  
emp\_id NUMBER PRIMARY KEY,  
name VARCHAR2(100),  
department\_id NUMBER,  
salary NUMBER(10, 2)  
);  
  
// Insert Sample Data into CUSTOMERS  
INSERT INTO customers VALUES (1, 'Alice', 65, 8000.00, 'FALSE');  
INSERT INTO customers VALUES (2, 'Bob', 45, 15000.00, 'FALSE');  
INSERT INTO customers VALUES (3, 'Charlie', 72, 12000.00, 'FALSE');  
INSERT INTO customers VALUES (4, 'Diana', 55, 9500.00, 'FALSE');  
  
// Insert Sample Data into LOANS  
INSERT INTO loans VALUES (101, 1, 6.5, SYSDATE + 15);  
INSERT INTO loans VALUES (102, 2, 7.2, SYSDATE + 10);  
INSERT INTO loans VALUES (103, 3, 6.8, SYSDATE + 45);  
  
// Insert Sample Data into SAVINGS\_ACCOUNTS  
INSERT INTO savings\_accounts VALUES (201, 1, 1000.00);  
INSERT INTO savings\_accounts VALUES (202, 2, 2500.00);  
  
// Insert Sample Data into ACCOUNTS  
INSERT INTO accounts VALUES (1001, 1, 2000.00);  
INSERT INTO accounts VALUES (1002, 2, 1500.00);  
  
// Insert Sample Data into EMPLOYEES  
INSERT INTO employees VALUES (301, 'John', 10, 3000.00);  
INSERT INTO employees VALUES (302, 'Sara', 10, 3500.00);

Scenario 1: The bank wants to apply a discount to loan interest rates for customers above 60 years old.

**Question:**

Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

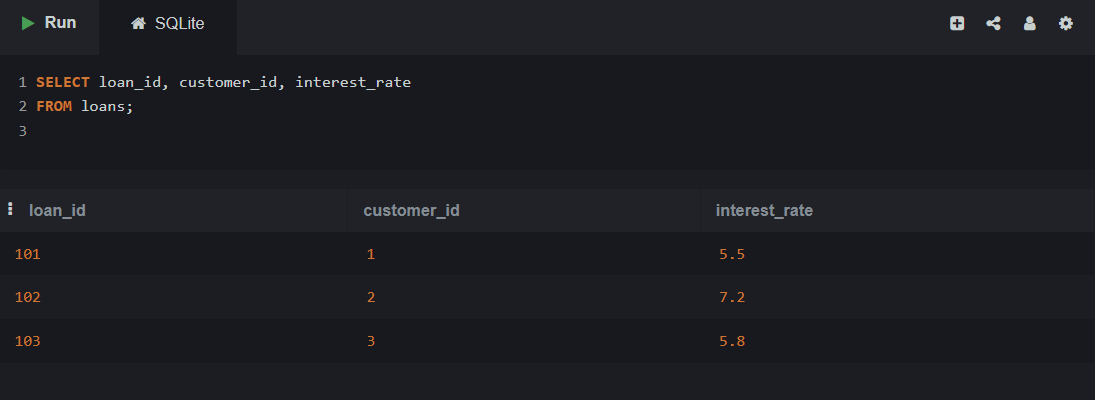
QUERY:

UPDATE loans

SET interest\_rate = interest\_rate - 1

WHERE customer\_id IN (

SELECT customer\_id FROM customers WHERE age > 60

);

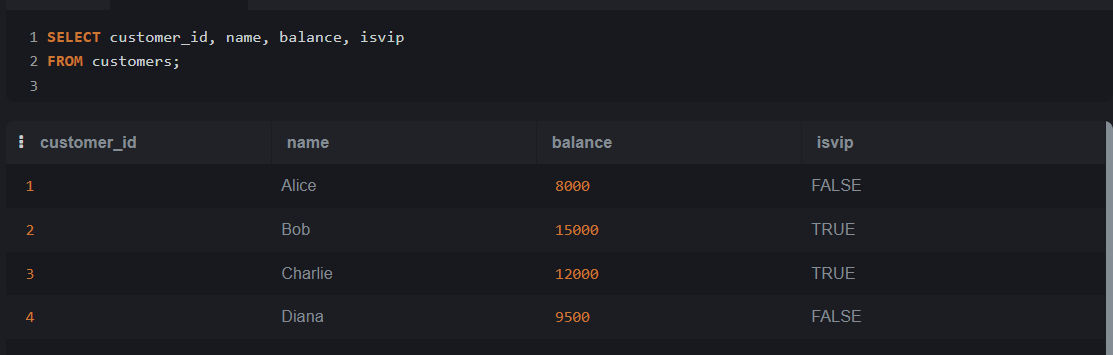
Scenario 2: A customer can be promoted to VIP status based on their balance.

**Question:**

Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.

**QUERY:**

BEGIN  
FOR customer\_rec IN (SELECT customer\_id, balance FROM customers) LOOP  
IF customer\_rec.balance > 10000 THEN  
UPDATE customers  
SET isvip = 'TRUE'  
WHERE customer\_id = customer\_rec.customer\_id;  
END IF;  
END LOOP;  
COMMIT;  
END;



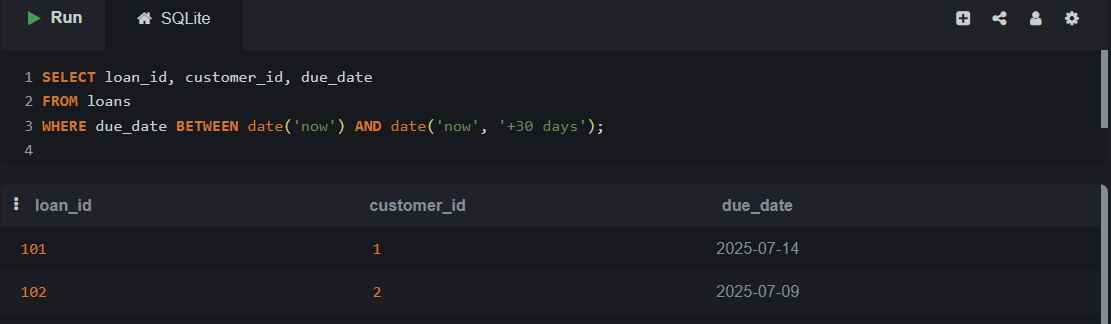
Scenario 3: The bank wants to send reminders to customers whose loans are due within the next 30 days.

**Question:**

**Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.**

**QUERY:**

BEGIN  
FOR loan\_rec IN (  
SELECT loan\_id, customer\_id, due\_date  
FROM loans  
WHERE due\_date BETWEEN SYSDATE AND SYSDATE + 30  
) LOOP  
DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ID ' || loan\_rec.loan\_id ||  
' for Customer ID ' || loan\_rec.customer\_id ||  
' is due on ' || TO\_CHAR(loan\_rec.due\_date, 'DD-MON-YYYY'));  
END LOOP;  
END;



**Exercise 3: Stored Procedures**

Scenario 1: The bank needs to process monthly interest for all savings accounts.

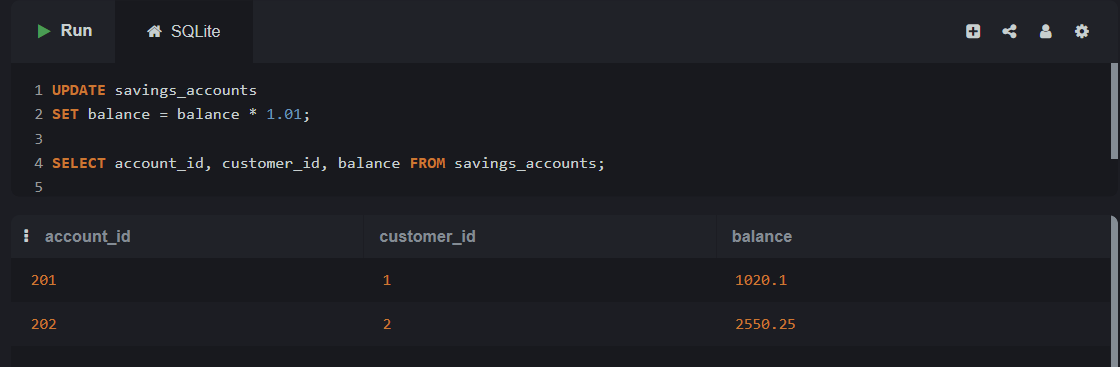
**Question:**

**Write a stored procedure ProcessMonthlyInterest that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.**

**QUERY:**

UPDATE savings\_accounts  
SET balance = balance \* 1.01;

SELECT account\_id, customer\_id, balance  
FROM savings\_accounts;



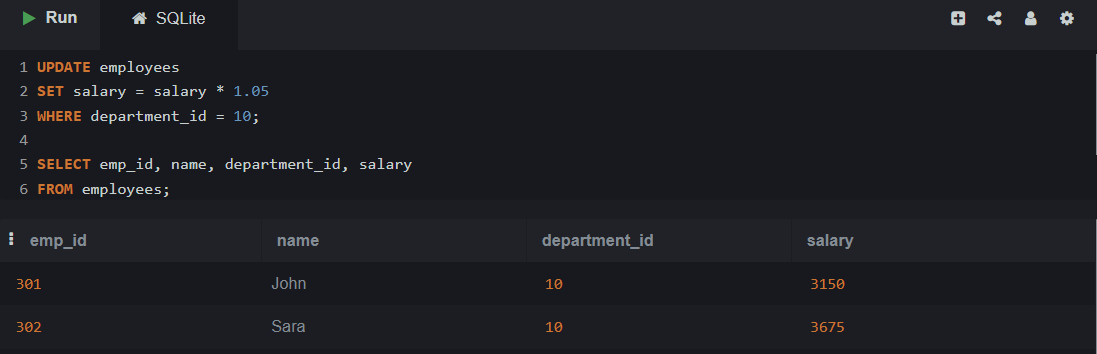
Scenario 2: The bank wants to implement a bonus scheme for employees based on their performance.

**Question:**

**Write a stored procedure UpdateEmployeeBonus that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.**

**QUERY:**

UPDATE employees  
SET salary = salary \* 1.05  
WHERE department\_id = 10;  
  
SELECT emp\_id, name, department\_id, salary  
FROM employees;



Scenario 3: Customers should be able to transfer funds between their accounts.

**Question:**

**Write a stored procedure TransferFunds that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.**

**QUERY:**

SELECT balance  
FROM accounts  
WHERE account\_id = 1001;  
UPDATE accounts  
SET balance = balance - 500  
WHERE account\_id = 1001  
AND balance >= 500;  
UPDATE accounts  
SET balance = balance + 500  
WHERE account\_id = 1002;  
SELECT account\_id, customer\_id, balance  
FROM accounts  
WHERE account\_id IN (1001, 1002);

