***PLSQL\_EXERCISES***

***Exercise 1: Control Structures***

* *Table Creation and Data Insert*

*BEGIN*

*EXECUTE IMMEDIATE 'DROP TABLE loans';*

*EXECUTE IMMEDIATE 'DROP TABLE customers';*

*EXCEPTION*

*WHEN OTHERS THEN NULL;*

*END;*

*/*

*CREATE TABLE customers (*

*customer\_id NUMBER PRIMARY KEY,*

*customer\_name VARCHAR2(100),*

*age NUMBER,*

*balance NUMBER(10,2),*

*is\_vip VARCHAR2(5) DEFAULT 'FALSE'*

*);*

*CREATE TABLE loans (*

*loan\_id NUMBER PRIMARY KEY,*

*customer\_id NUMBER,*

*loan\_amount NUMBER(10,2),*

*interest\_rate NUMBER(5,2),*

*due\_date DATE,*

*FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)*

*);*

*INSERT INTO customers VALUES (1, 'John Smith', 65, 15000.00, 'FALSE');*

*INSERT INTO customers VALUES (2, 'Mary Johnson', 45, 8500.00, 'FALSE');*

*INSERT INTO customers VALUES (3, 'Robert Brown', 72, 25000.00, 'FALSE');*

*INSERT INTO customers VALUES (4, 'Sarah Davis', 55, 12000.00, 'FALSE');*

*INSERT INTO customers VALUES (5, 'Michael Wilson', 68, 7500.00, 'FALSE');*

*INSERT INTO loans VALUES (1, 1, 50000.00, 5.5, DATE '2025-07-15');*

*INSERT INTO loans VALUES (2, 2, 30000.00, 6.0, DATE '2025-07-20');*

*INSERT INTO loans VALUES (3, 3, 75000.00, 5.0, DATE '2025-08-10');*

*INSERT INTO loans VALUES (4, 4, 40000.00, 5.8, DATE '2025-07-25');*

*INSERT INTO loans VALUES (5, 5, 20000.00, 6.2, DATE '2025-07-05');*

*COMMIT;*

* ***PL/SQL Code for All Scenarios***

*Scenario 1: Interest Discount*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE('=== Interest Rate Discount for Customers Above 60 ===');*

*FOR rec IN (*

*SELECT c.customer\_id, c.customer\_name, c.age, l.loan\_id, l.interest\_rate*

*FROM customers c*

*JOIN loans l ON c.customer\_id = l.customer\_id*

*WHERE c.age > 60*

*) LOOP*

*UPDATE loans*

*SET interest\_rate = interest\_rate - 1*

*WHERE loan\_id = rec.loan\_id;*

*DBMS\_OUTPUT.PUT\_LINE('Customer: ' || rec.customer\_name ||*

*', Loan ID: ' || rec.loan\_id ||*

*', New Rate: ' || (rec.interest\_rate - 1));*

*END LOOP;*

*COMMIT;*

*END;*

*/*

*Scenario 2: Promote VIP*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE(CHR(10) || '=== Promoting Customers to VIP (Balance > $10,000) ===');*

*FOR rec IN (SELECT customer\_id, customer\_name, balance FROM customers WHERE balance > 10000) LOOP*

*UPDATE customers SET is\_vip = 'TRUE' WHERE customer\_id = rec.customer\_id;*

*DBMS\_OUTPUT.PUT\_LINE('Customer: ' || rec.customer\_name ||*

*' promoted to VIP (Balance: $' || TO\_CHAR(rec.balance, '999,999.00') || ')');*

*END LOOP;*

*COMMIT;*

*END;*

*/*

*Scenario 3: Payment Reminders*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE(CHR(10) || '=== Loan Payment Reminders (Due within 30 days) ===');*

*DBMS\_OUTPUT.PUT\_LINE('Current Date: ' || TO\_CHAR(SYSDATE, 'DD-MON-YYYY'));*

*FOR rec IN (*

*SELECT l.loan\_id, c.customer\_name, l.loan\_amount, l.due\_date,*

*(l.due\_date - SYSDATE) AS days\_until\_due*

*FROM loans l JOIN customers c ON l.customer\_id = c.customer\_id*

*WHERE l.due\_date BETWEEN SYSDATE AND SYSDATE + 30*

*ORDER BY l.due\_date*

*) LOOP*

*DBMS\_OUTPUT.PUT\_LINE(CHR(10) || 'REMINDER: Dear ' || rec.customer\_name || ',');*

*DBMS\_OUTPUT.PUT\_LINE('Your loan of $' || TO\_CHAR(rec.loan\_amount, '999,999.00') ||*

*' is due on ' || TO\_CHAR(rec.due\_date, 'DD-MON-YYYY') ||*

*' (' || ROUND(rec.days\_until\_due) || ' days remaining)');*

*IF ROUND(rec.days\_until\_due) <= 7 THEN*

*DBMS\_OUTPUT.PUT\_LINE('\*\*\* URGENT - Due within a week! \*\*\*');*

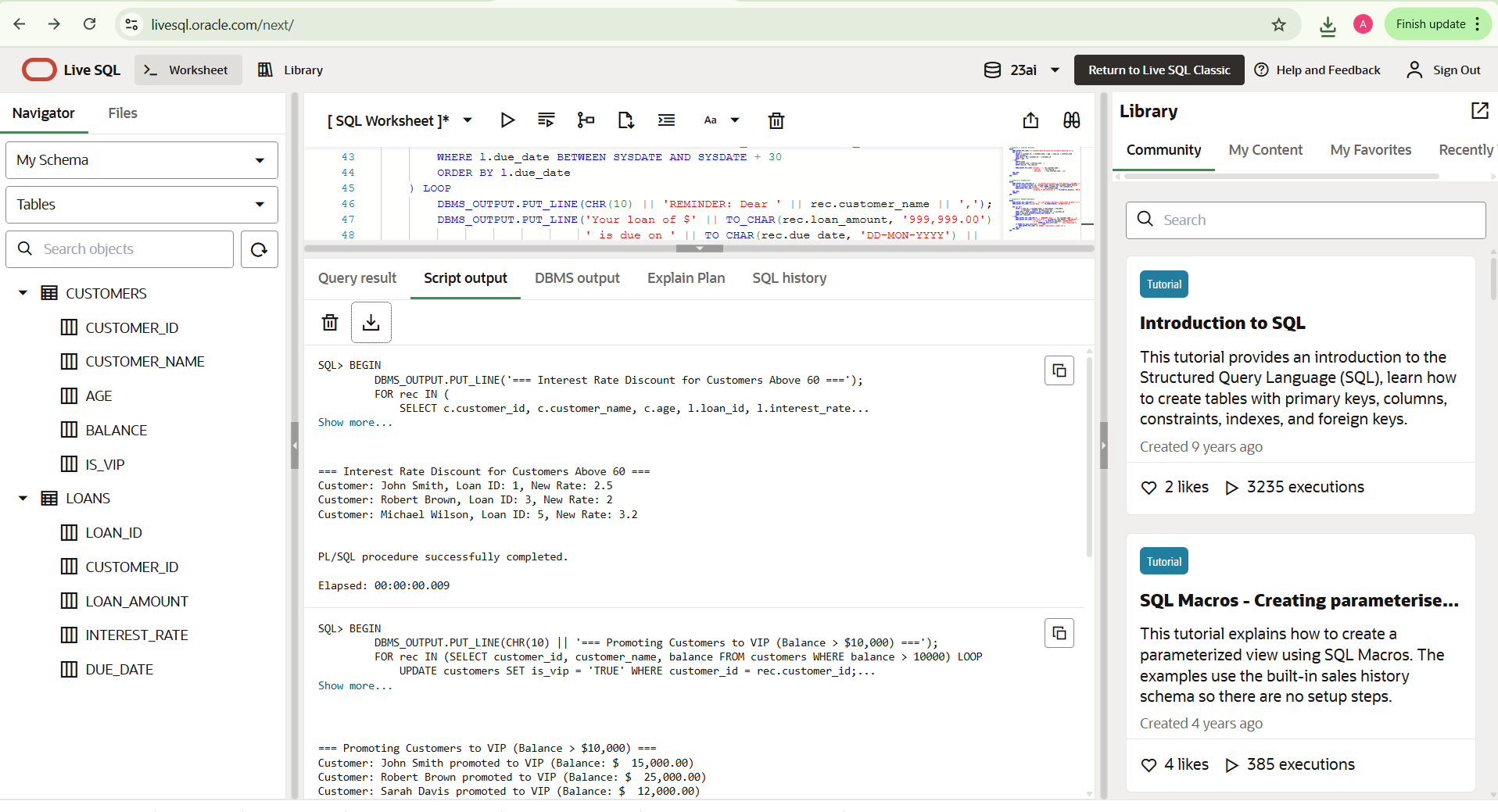
*END IF;*

*END LOOP;*

*END;*

*/*

***OUTPUT:***

******

***A screenshot of a computer

AI-generated content may be incorrect.***

***Exercise 3: Stored Procedures***

* *Table Creation and Data Insert*

*CREATE TABLE accounts (*

*account\_id NUMBER PRIMARY KEY,*

*customer\_name VARCHAR2(100),*

*account\_type VARCHAR2(20),*

*balance NUMBER(10,2)*

*);*

*CREATE TABLE employees (*

*emp\_id NUMBER PRIMARY KEY,*

*emp\_name VARCHAR2(100),*

*department VARCHAR2(50),*

*salary NUMBER(10,2)*

*);*

*INSERT INTO accounts VALUES (1, 'John Smith', 'Savings', 10000);*

*INSERT INTO accounts VALUES (2, 'Mary Johnson', 'Savings', 12000);*

*INSERT INTO accounts VALUES (3, 'Robert Brown', 'Current', 8000);*

*INSERT INTO accounts VALUES (4, 'Sarah Davis', 'Savings', 15000);*

*INSERT INTO employees VALUES (101, 'Alice', 'Sales', 50000);*

*INSERT INTO employees VALUES (102, 'Bob', 'Sales', 55000);*

*INSERT INTO employees VALUES (103, 'Charlie', 'HR', 48000);*

*COMMIT;*

* ***PL/SQL Code for All Scenarios***

*CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE('=== Monthly Interest Process ===');*

*FOR acc IN (SELECT account\_id, balance FROM accounts WHERE account\_type = 'Savings') LOOP*

*UPDATE accounts*

*SET balance = balance + (balance \* 0.01)*

*WHERE account\_id = acc.account\_id;*

*DBMS\_OUTPUT.PUT\_LINE('Interest added to Account ID: ' || acc.account\_id);*

*END LOOP;*

*COMMIT;*

*END;*

*/*

*CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (*

*dept\_name IN VARCHAR2,*

*bonus\_percent IN NUMBER*

*) AS*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE('=== Employee Bonus Update ===');*

*FOR emp IN (SELECT emp\_id FROM employees WHERE department = dept\_name) LOOP*

*UPDATE employees*

*SET salary = salary + (salary \* bonus\_percent / 100)*

*WHERE emp\_id = emp.emp\_id;*

*DBMS\_OUTPUT.PUT\_LINE('Bonus applied to Employee ID: ' || emp.emp\_id);*

*END LOOP;*

*COMMIT;*

*END;*

*/*

*CREATE OR REPLACE PROCEDURE TransferFunds (*

*from\_acc\_id IN NUMBER,*

*to\_acc\_id IN NUMBER,*

*amount IN NUMBER*

*) AS*

*from\_balance NUMBER;*

*BEGIN*

*SELECT balance INTO from\_balance FROM accounts WHERE account\_id = from\_acc\_id;*

*IF from\_balance < amount THEN*

*RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account');*

*ELSE*

*UPDATE accounts SET balance = balance - amount WHERE account\_id = from\_acc\_id;*

*UPDATE accounts SET balance = balance + amount WHERE account\_id = to\_acc\_id;*

*DBMS\_OUTPUT.PUT\_LINE('Transferred ₹' || amount || ' from A/c ' || from\_acc\_id || ' to A/c ' || to\_acc\_id);*

*END IF;*

*COMMIT;*

*END;*

*/*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE('=== RUNNING ALL 3 SCENARIOS ===');*

*ProcessMonthlyInterest;*

*UpdateEmployeeBonus('Sales', 10);*

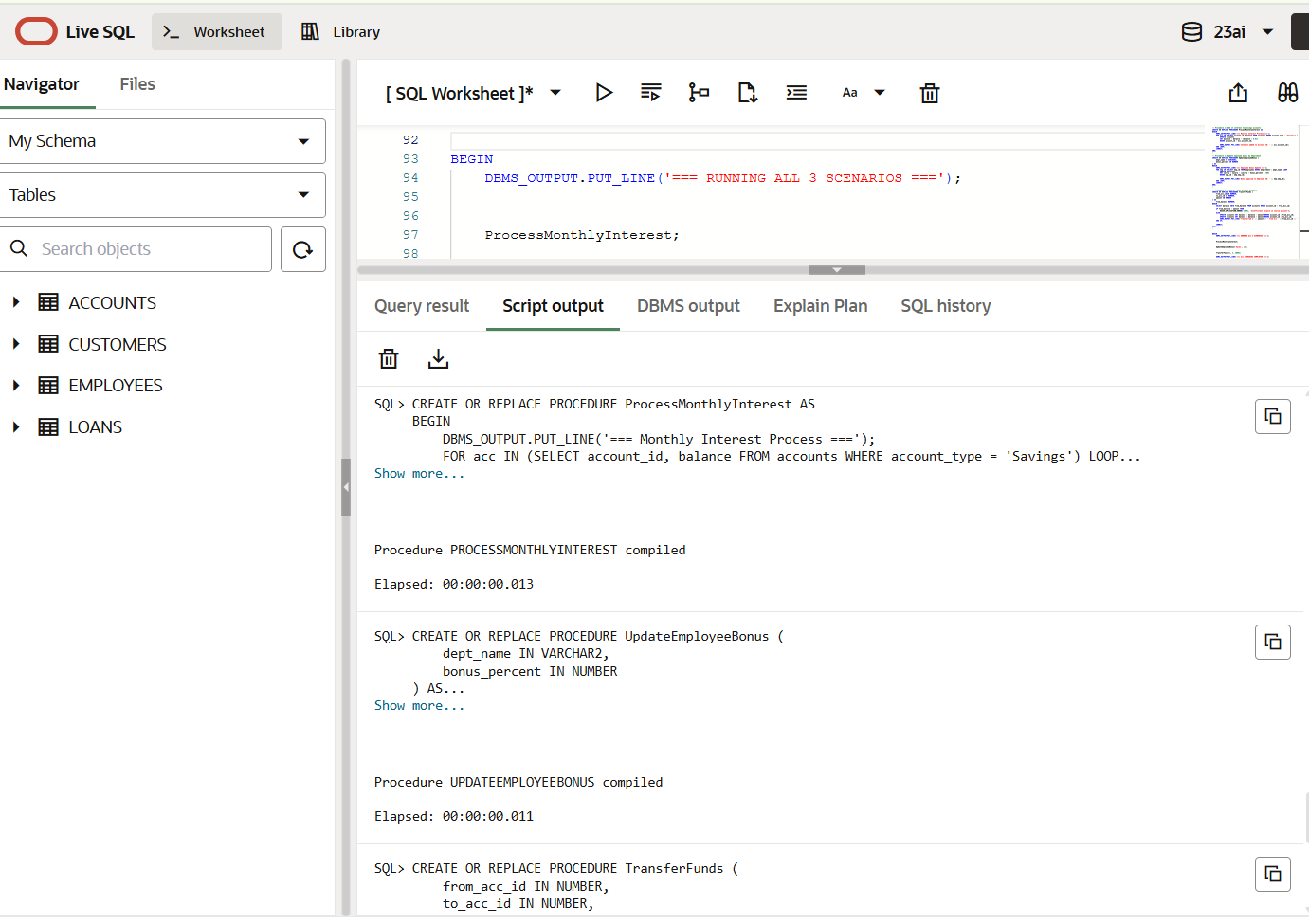
*TransferFunds(1, 3, 2000);*

*DBMS\_OUTPUT.PUT\_LINE('=== ALL SCENARIOS COMPLETED ===');*

*END;*

*/*

***OUTPUT:***

******

***A screenshot of a computer

AI-generated content may be incorrect.***