**Exercise 6: Cursors**

**Scenario 1:** Generate monthly statements for all customers.

* **Question:** Write a PL/SQL block using an explicit cursor **GenerateMonthlyStatements** that retrieves all transactions for the current month and prints a statement for each customer.

**QUERY:**

SET SERVEROUTPUT ON;

DECLARE

CURSOR GenerateMonthlyStatements IS

SELECT customer\_id, SUM(amount) AS total\_amount

FROM Transactions

WHERE EXTRACT(MONTH FROM transaction\_date) = EXTRACT(MONTH FROM SYSDATE)

GROUP BY customer\_id;

v\_customer\_id Transactions.customer\_id%TYPE;

v\_total\_amount Transactions.amount%TYPE;

BEGIN

FOR statement IN GenerateMonthlyStatements LOOP

v\_customer\_id := statement.customer\_id;

v\_total\_amount := statement.total\_amount;

DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_customer\_id);

DBMS\_OUTPUT.PUT\_LINE('Total Amount for the Month: ' || v\_total\_amount);

DBMS\_OUTPUT.PUT\_LINE('----------------------------------');

END LOOP;

END;

**Explanation:**

The cursor `GenerateMonthlyStatements` retrieves the `customer\_id` and the total `amount` of transactions for the current month from the `Transactions` table. The block then iterates over each customer's total monthly transactions, printing out a statement for each customer showing their `customer\_id` and the `total\_amount` for the month.

**Scenario 2:** Apply annual fee to all accounts.

* **Question:** Write a PL/SQL block using an explicit cursor **ApplyAnnualFee** that deducts an annual maintenance fee from the balance of all accounts.

**QUERY:**

SET SERVEROUTPUT ON;

DECLARE

CURSOR ApplyAnnualFee IS

SELECT account\_id, balance

FROM Accounts;

v\_account\_id Accounts.account\_id%TYPE;

v\_balance Accounts.balance%TYPE;

v\_annual\_fee CONSTANT NUMBER := 50; -- Assuming the annual maintenance fee is $50

BEGIN

FOR account\_info IN ApplyAnnualFee LOOP

v\_account\_id := account\_info.account\_id;

v\_balance := account\_info.balance;

UPDATE Accounts

SET balance = balance - v\_annual\_fee

WHERE account\_id = v\_account\_id;

DBMS\_OUTPUT.PUT\_LINE('Annual fee of $50 deducted from Account ID: ' || v\_account\_id);

END LOOP;

COMMIT; -- Commit the changes to the database

END;

**Explanation:**

The cursor `ApplyAnnualFee` retrieves the `account\_id` and `balance` for all accounts from the `Accounts` table. The block then iterates over each account, deducting the annual maintenance fee (assumed to be $50) from the account's balance using an `UPDATE` statement. After applying the fee to all accounts, the changes are committed to the database using `COMMIT`.

**Scenario 3:** Update the interest rate for all loans based on a new policy.

* **Question:** Write a PL/SQL block using an explicit cursor **UpdateLoanInterestRates** that fetches all loans and updates their interest rates based on the new policy.

**QUERY:**

SET SERVEROUTPUT ON;

DECLARE

CURSOR UpdateLoanInterestRates IS

SELECT loan\_id, interest\_rate

FROM Loans;

v\_loan\_id Loans.loan\_id%TYPE;

v\_interest\_rate Loans.interest\_rate%TYPE;

v\_new\_policy\_interest\_rate CONSTANT NUMBER := 0.05; -- Assuming the new policy sets the interest rate to 5%

BEGIN

FOR loan\_info IN UpdateLoanInterestRates LOOP

v\_loan\_id := loan\_info.loan\_id;

v\_interest\_rate := loan\_info.interest\_rate;

UPDATE Loans

SET interest\_rate = v\_new\_policy\_interest\_rate

WHERE loan\_id = v\_loan\_id;

DBMS\_OUTPUT.PUT\_LINE('Interest rate updated for Loan ID: ' || v\_loan\_id || ' to ' || v\_new\_policy\_interest\_rate);

END LOOP;

COMMIT; -- Commit the changes to the database

END;

**Explanation:**

The cursor `UpdateLoanInterestRates` fetches the `loan\_id` and current `interest\_rate` for all loans from the `Loans` table. The block then iterates over each loan, updating the interest rate to the new policy rate (assumed to be 5%) using an `UPDATE` statement. After updating the interest rates for all loans, the changes are committed to the database using `COMMIT`.