SuperSet ID : 6314413 **Shaik Balaji Mahammad Rafi**

HANDSON EXERCISES - WEEK 2

Skill : PL/SQL

Exercise 1: Control Structures

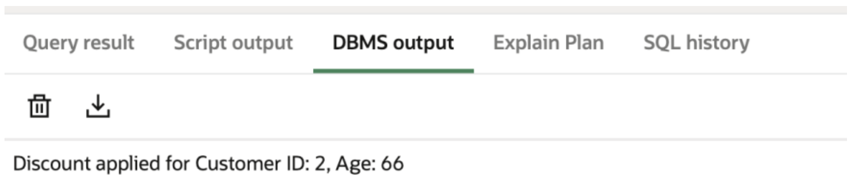
Scenario 1: The bank wants to apply a discount to loan interest rates for customers above 60 years old.

Question 1 : 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.

CODE :

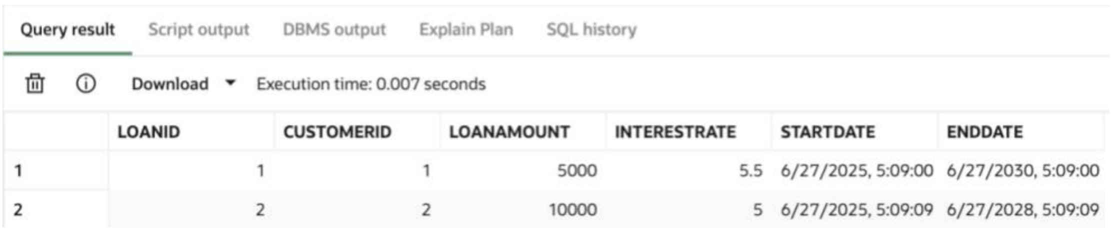
```
BEGIN
FOR cust IN (SELECT CustomerID, DOB FROM Customers) LOOP
DECLARE
v_age NUMBER;
BEGIN

SELECT FLOOR(MONTHS_BETWEEN(SYSDATE, cust.DOB) / 12)
INTO v_age
FROM dual;

IF v_age > 60 THEN
UPDATE Loans
SET InterestRate = InterestRate - 1
WHERE CustomerID = cust.CustomerID;

DBMS_OUTPUT.PUT_LINE('Discount applied for Customer ID: ' || cust.CustomerID || ', Age: '
|| v_age);
END IF;
END;
END LOOP;

COMMIT;
END;
/
```

OUTPUT :



```
Discount applied for Customer ID: 2, Age: 66
```

```
SELECT * FROM Loans;
```


Query result Script output DBMS output Explain Plan SQL history

Download Execution time: 0.007 seconds

	LOANID	CUSTOMERID	LOANAMOUNT	INTERESTRATE	STARTDATE	ENDDATE
1	1	1	5000	5.5	6/27/2025, 5:09:00	6/27/2030, 5:09:00
2	2	2	10000	5	6/27/2025, 5:09:09	6/27/2028, 5:09:09

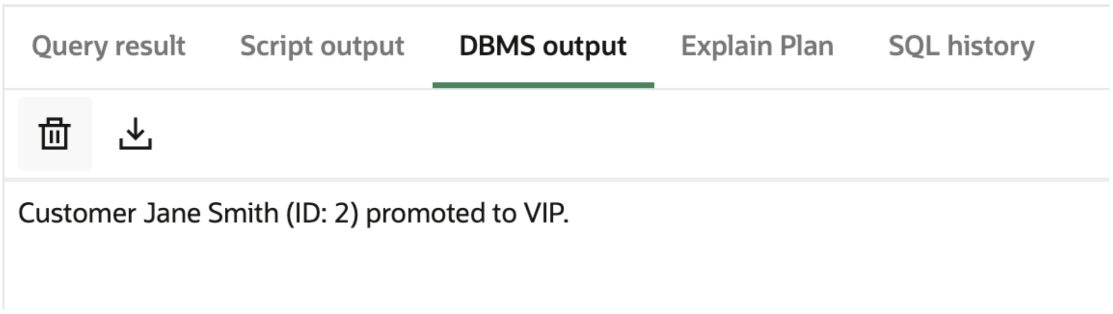
Scenario 2 : A customer can be promoted to VIP status based on their balance.

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.

CODE :

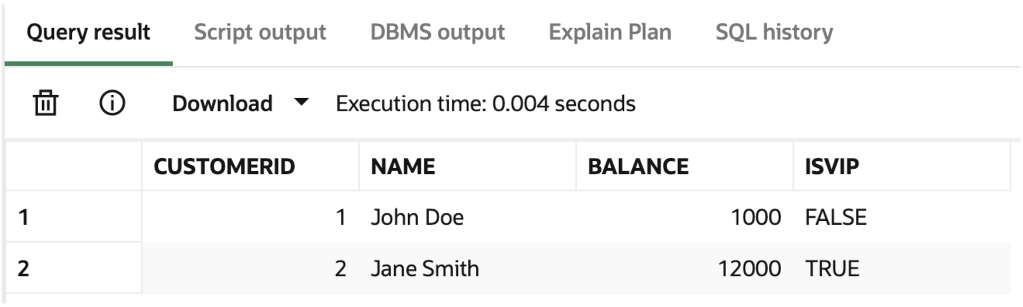
```
BEGIN
FOR cust IN (SELECT CustomerID, Name, Balance FROM Customers) LOOP
IF cust.Balance > 10000 THEN
UPDATE Customers
SET IsVIP = 'TRUE'
WHERE CustomerID = cust.CustomerID;

DBMS_OUTPUT.PUT_LINE('Customer ' || cust.Name || ' (ID: ' || cust.CustomerID || ')
promoted to VIP.');
ELSE
UPDATE Customers
SET IsVIP = 'FALSE'
WHERE CustomerID = cust.CustomerID;
END IF;
END LOOP;

COMMIT;
END;
/
```

OUTPUT :



```
Customer Jane Smith (ID: 2) promoted to VIP.
```

```
SELECT CustomerID, Name, Balance, IsVIP FROM Customers;
```



	CUSTOMERID	NAME	BALANCE	ISVIP
1	1	John Doe	1000	FALSE
2	2	Jane Smith	12000	TRUE

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.

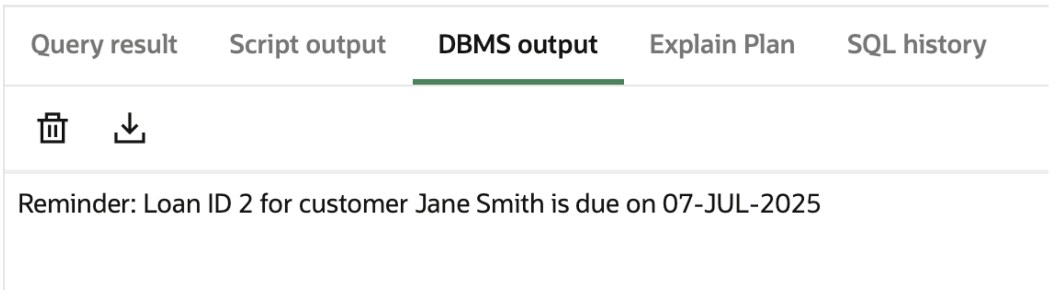
CODE :

```
UPDATE Loans SET EndDate = SYSDATE + 10 WHERE LoanID = 2;
```

After updating ,

```
BEGIN

FOR loan_rec IN (
SELECT l.LoanID, l.CustomerID, l.EndDate, c.Name
FROM Loans l
JOIN Customers c ON l.CustomerID = c.CustomerID
WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30
) LOOP
DBMS_OUTPUT.PUT_LINE(
'Reminder: Loan ID ' || loan_rec.LoanID || ' for customer ' || loan_rec.Name ||
' is due on ' || TO_CHAR(loan_rec.EndDate, 'DD-MON-YYYY')
);
END LOOP;
END;
/
```

OUTPUT :



```
Reminder: Loan ID 2 for customer Jane Smith is due on 07-JUL-2025
```

Exercise 3: Stored Procedures

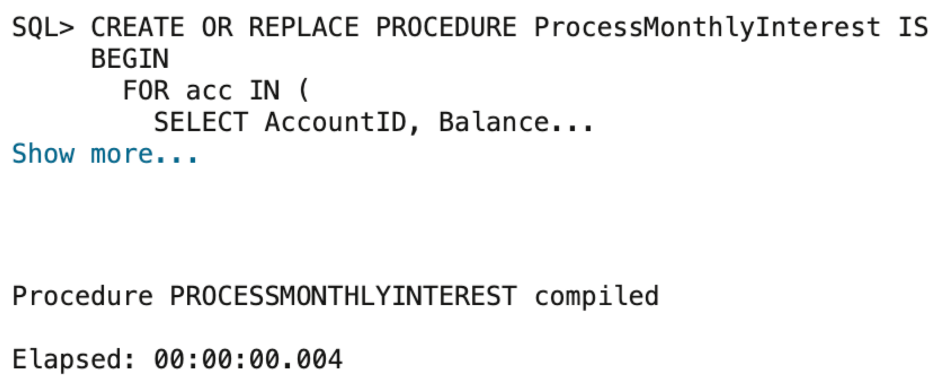
Scenario 1 : The bank needs to process monthly interest for all savings accounts.

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.

CODE :

Creating a Procedure

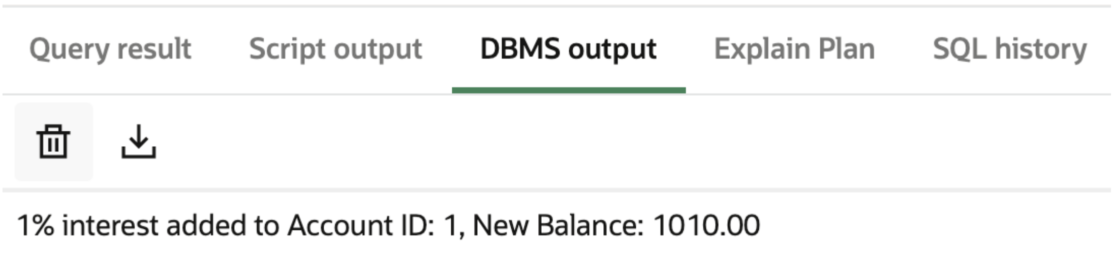
```
CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS
BEGIN
FOR acc IN (
SELECT AccountID, Balance
FROM Accounts
WHERE AccountType = 'Savings'
) LOOP
UPDATE Accounts

SET Balance = Balance + (acc.Balance * 0.01)
WHERE AccountID = acc.AccountID;

DBMS_OUTPUT.PUT_LINE(
'1% interest added to Account ID: ' || acc.AccountID ||
', New Balance: ' || TO_CHAR(acc.Balance * 1.01, '9999.99')
);
END LOOP;

COMMIT;
END;
/
```



```
SQL> CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS
     BEGIN
       FOR acc IN (
         SELECT AccountID, Balance...
Show more...


Procedure PROCESSMONTHLYINTEREST compiled

Elapsed: 00:00:00.004
```

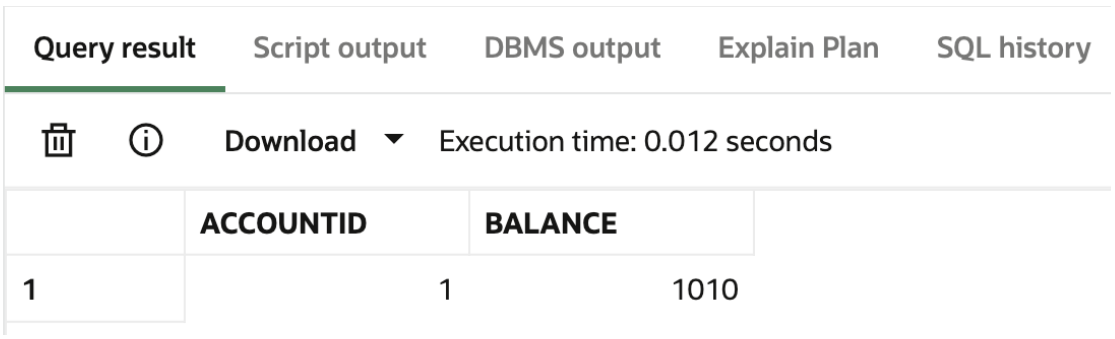
Running the Procedure :

```
BEGIN
ProcessMonthlyInterest;
END;
/
```



```
1% interest added to Account ID: 1, New Balance: 1010.00
```

```
SELECT AccountID, Balance FROM Accounts WHERE AccountID = 1;
```



	ACCOUNTID	BALANCE
1	1	1010

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

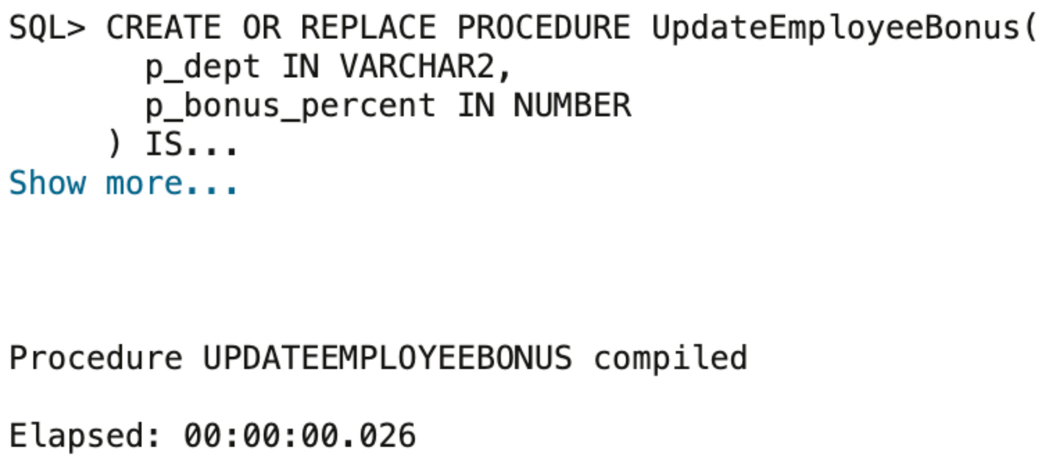
Question: Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

CODE :

Creating a Procedure

```
CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(
p_dept IN VARCHAR2,
p_bonus_percent IN NUMBER
) IS
BEGIN
FOR emp IN (
SELECT EmployeeID, Name, Salary
FROM Employees
WHERE Department = p_dept
) LOOP
```

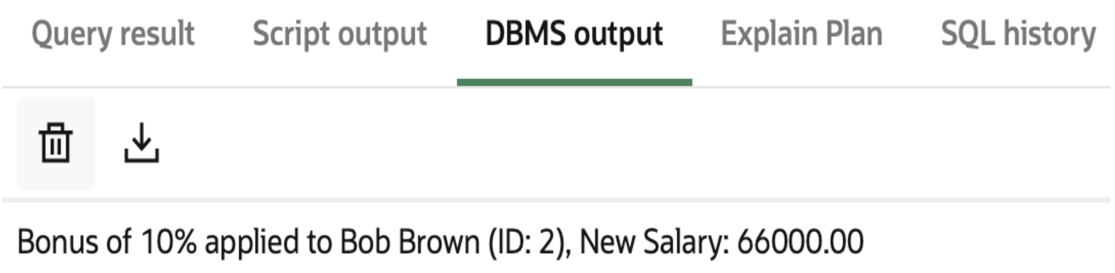
```
UPDATE Employees
SET Salary = Salary + (emp.Salary * (p_bonus_percent / 100))
WHERE EmployeeID = emp.EmployeeID;

DBMS_OUTPUT.PUT_LINE(
'Bonus of ' || p_bonus_percent || '% applied to ' || emp.Name ||
' (ID: ' || emp.EmployeeID || '), New Salary: ' ||
TO_CHAR(emp.Salary * (1 + p_bonus_percent / 100), '999999.99')
);
END LOOP;

COMMIT;
END;
/
```



```
SQL> CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(
       p_dept IN VARCHAR2,
       p_bonus_percent IN NUMBER
     ) IS...
Show more...


Procedure UPDATEEMPLOYEEBONUS compiled

Elapsed: 00:00:00.026
```

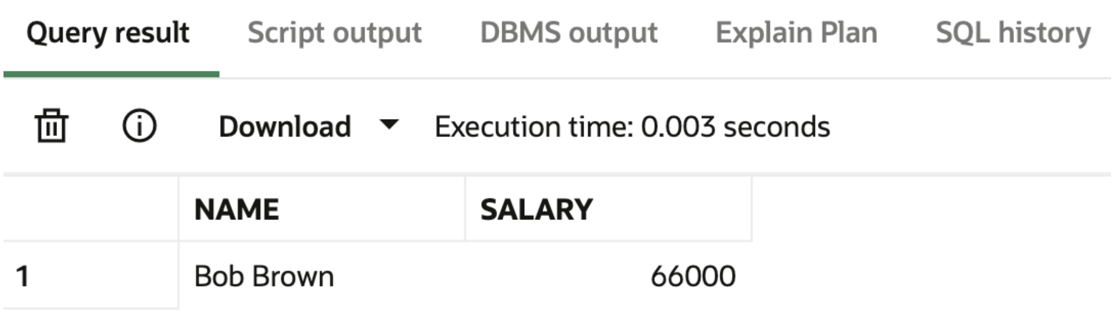
Running the Procedure :

```
BEGIN
UpdateEmployeeBonus('IT', 10);
END;
/
```



```
Bonus of 10% applied to Bob Brown (ID: 2), New Salary: 66000.00
```

```
SELECT Name, Salary FROM Employees WHERE EmployeeID = 2;
```



	NAME	SALARY
1	Bob Brown	66000

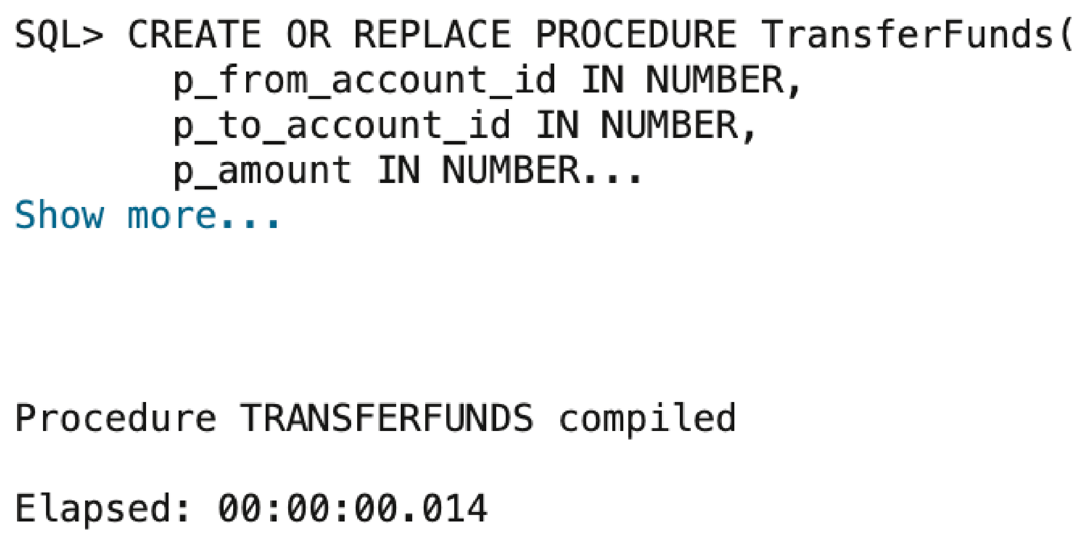
Scenario 3 : Customers should be able to transfer funds between their accounts.

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.

CODE :
Creating a Procedure

```
CREATE OR REPLACE PROCEDURE TransferFunds(
p_from_account_id IN NUMBER,
p_to_account_id IN NUMBER,
p_amount IN NUMBER
) IS
v_from_balance NUMBER;
BEGIN

SELECT Balance INTO v_from_balance
FROM Accounts
WHERE AccountID = p_from_account_id;

IF v_from_balance >= p_amount THEN

UPDATE Accounts
SET Balance = Balance - p_amount
WHERE AccountID = p_from_account_id;


UPDATE Accounts
SET Balance = Balance + p_amount
WHERE AccountID = p_to_account_id;
DBMS_OUTPUT.PUT_LINE('Transfer of ₹' || p_amount || ' successful from Account ' ||
p_from_account_id || ' to Account ' || p_to_account_id);
```

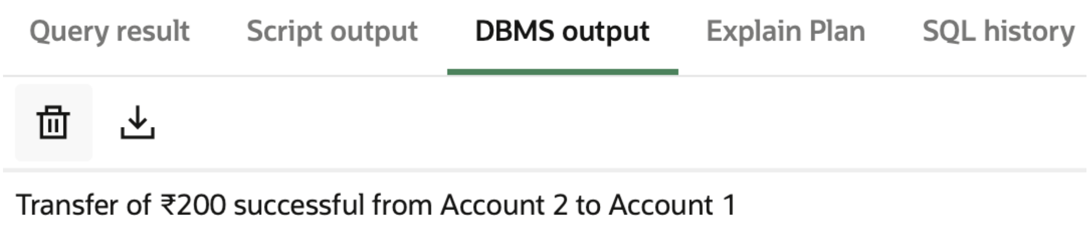
```
ELSE
DBMS_OUTPUT.PUT_LINE('Insufficient funds in Account ' || p_from_account_id);
END IF;

COMMIT;
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('One of the accounts does not exist.');
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE('An unexpected error occurred: ' || SQLERRM);
END;
/
```



```
SQL> CREATE OR REPLACE PROCEDURE TransferFunds(
       p_from_account_id IN NUMBER,
       p_to_account_id IN NUMBER,
       p_amount IN NUMBER...
Show more...


Procedure TRANSFERFUNDS compiled

Elapsed: 00:00:00.014
```

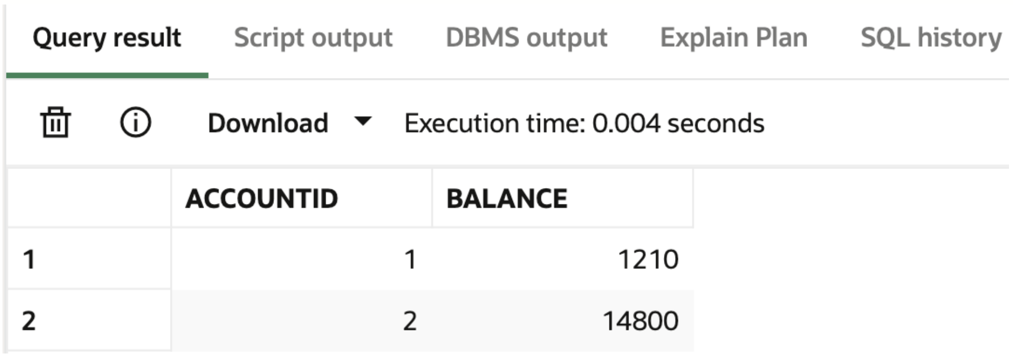
Running the Procedure :

```
BEGIN
TransferFunds(2, 1, 200);
END;
/
```



```
Transfer of ₹200 successful from Account 2 to Account 1
```

```
SELECT AccountID, Balance FROM Accounts;
```



Execution time: 0.004 seconds

	ACCOUNTID	BALANCE
1	1	1210
2	2	14800