Exercise 3 – Stored Procedures

# 1. Objective

To develop PL/SQL stored procedures that perform data manipulation tasks involving interest processing and employee bonus allocation.

# 2. Problem Statement / Scenario

You are working on automation tasks for a bank. Two key procedures are required:  
- Apply 1% interest to all savings account balances.  
- Update employee salaries by a bonus percentage for a specified department.

# 3. Approach / Steps

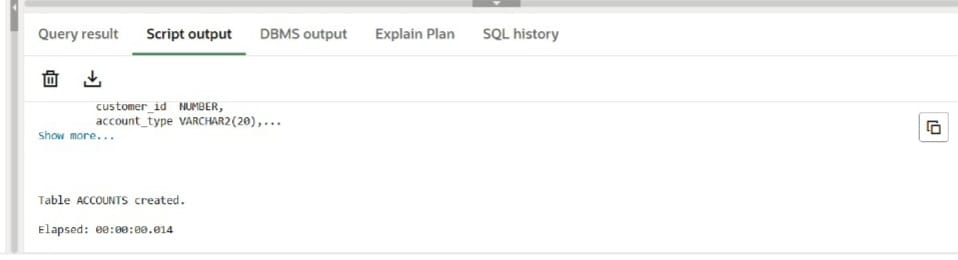
## 3.1 Understanding Stored Procedures

Stored procedures encapsulate logic that can be reused and executed with parameters. They are efficient for business logic that must be performed repeatedly or conditionally.

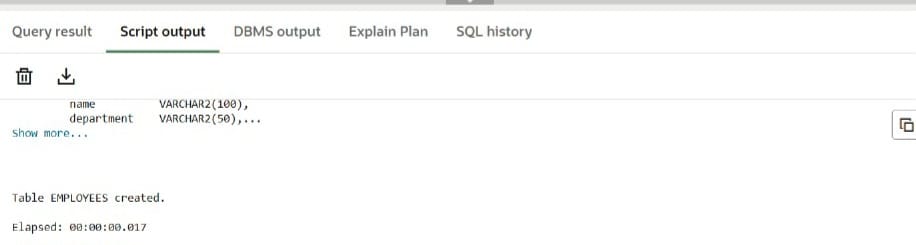
## 3.2 Setup

Create `accounts` and `employees` tables and insert test data.

Accounts Table Creation:



Employees Table Creation:



## 3.3 Implementation

Create two stored procedures:  
- `ProcessMonthlyInterest` to update balances  
- `UpdateEmployeeBonus` to increase salary

# 4. Code

Stored Procedure: ProcessMonthlyInterest

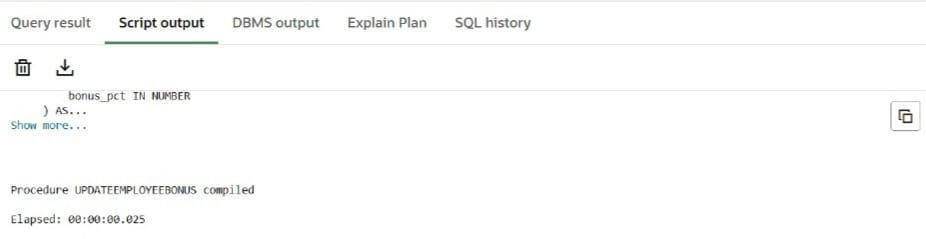
CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS  
BEGIN  
 UPDATE accounts  
 SET balance = balance \* 1.01  
 WHERE account\_type = 'Savings';  
   
 COMMIT;  
END;

Stored Procedure: UpdateEmployeeBonus

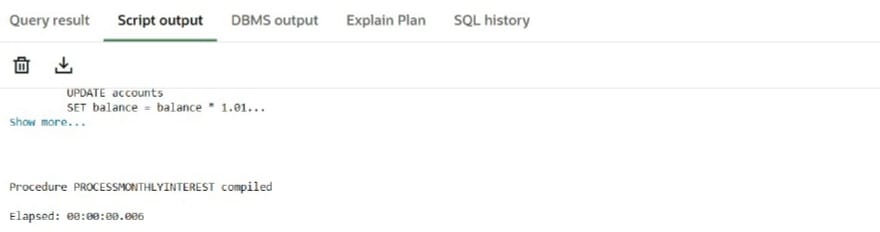
CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (  
 dept\_name IN VARCHAR2,  
 bonus\_pct IN NUMBER  
) AS  
BEGIN  
 UPDATE employees  
 SET salary = salary + (salary \* bonus\_pct / 100)  
 WHERE department = dept\_name;  
  
 COMMIT;  
END;

# 5. Output Verification

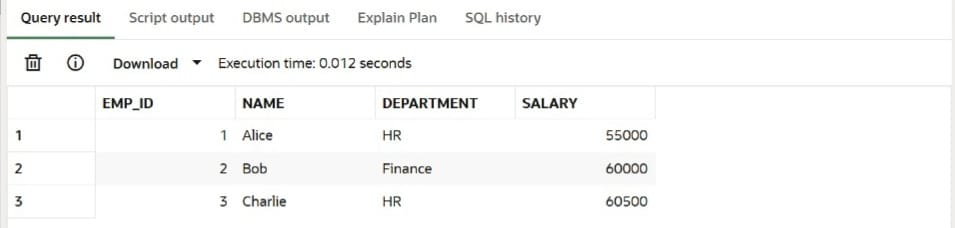
Procedure Compilation: UpdateEmployeeBonus



Procedure Compilation: ProcessMonthlyInterest



Updated Employee Salaries (After Applying Bonus):



# 6. Conclusion

Stored procedures provide a powerful way to encapsulate logic and apply consistent business rules. This exercise demonstrated how to use procedures to apply monthly interest to account balances and give conditional bonuses to employees.