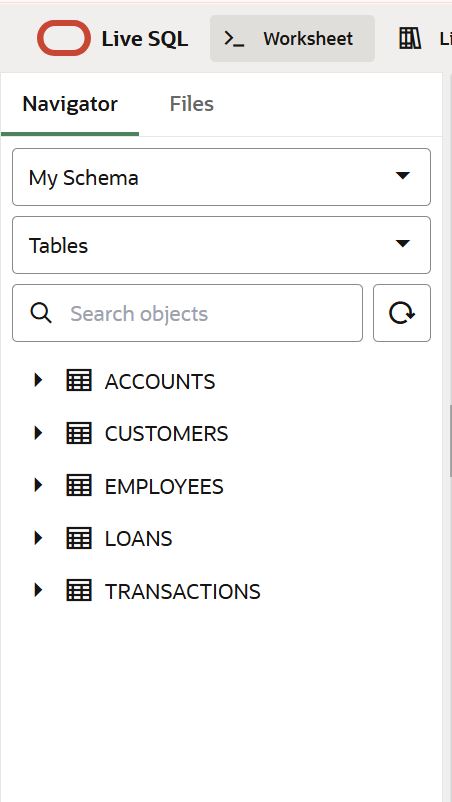
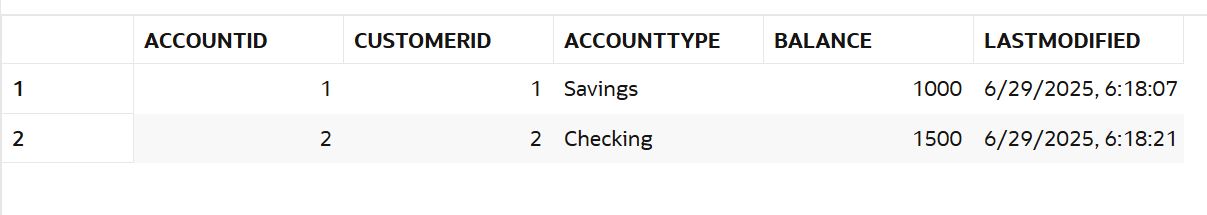
Arshitha Yasmin S

Superset Id:6400626

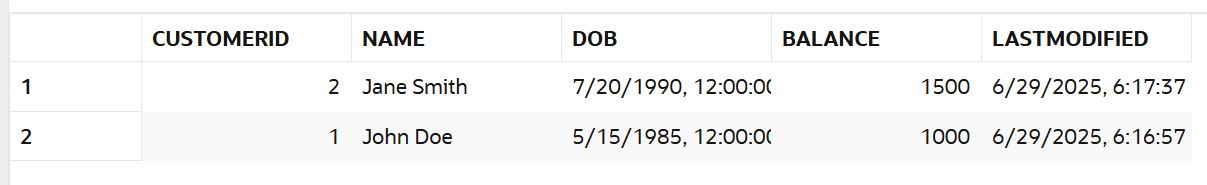
PL/SQL Solutions

I used Oracle to run the query, created the table provided in the Word document, and inserted the rows and columns as specified.

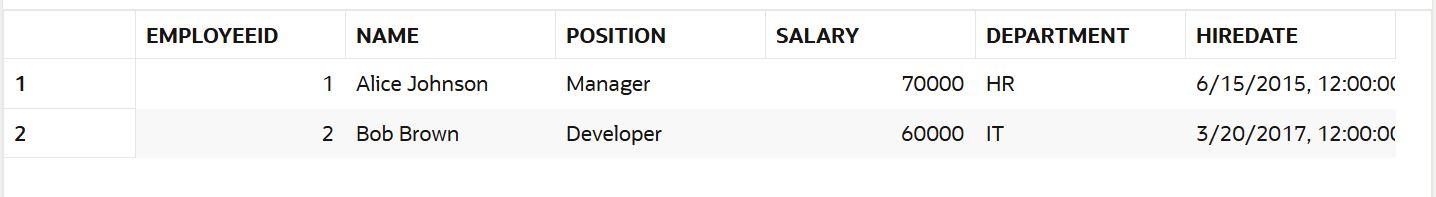


Account table:  


Customers table:



Employees table:



Loan table:

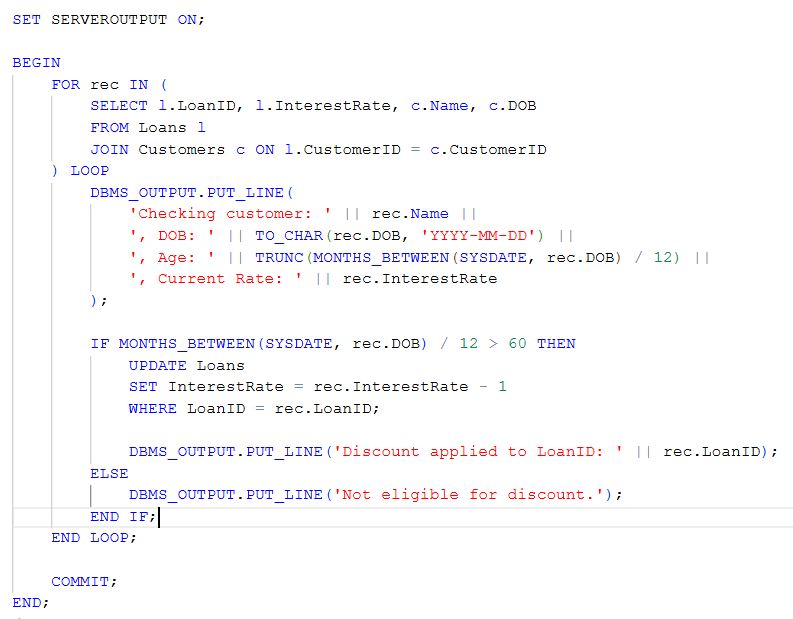


Transaction table:

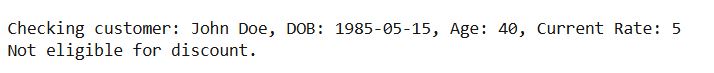


**Exercise 1: Control Structures**

**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.**



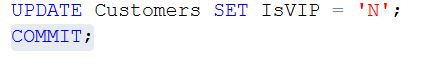
OUTPUT:



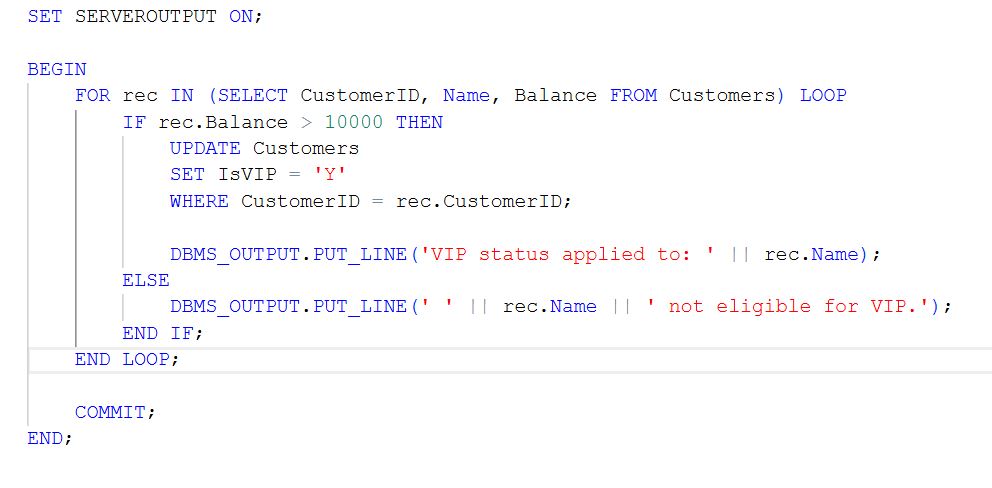


**2) Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.**

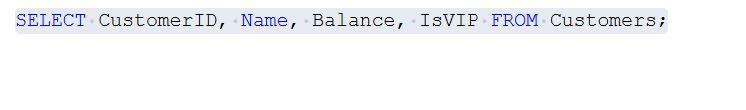
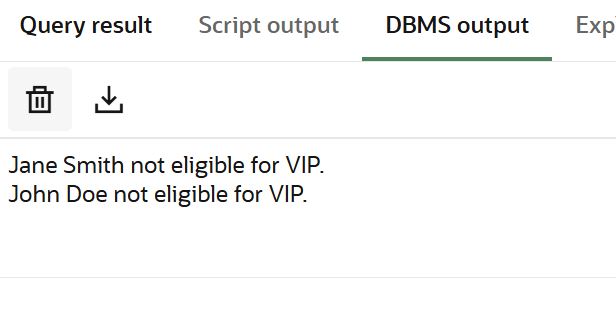
****

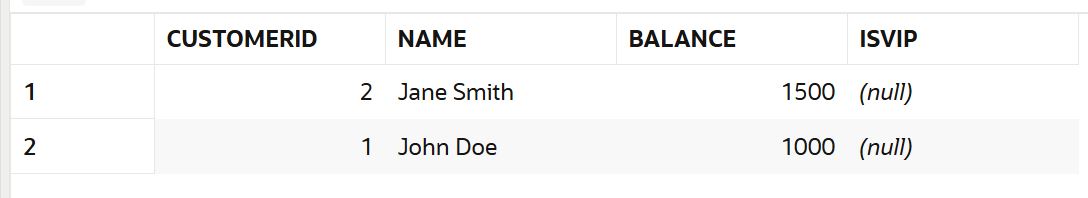
****

**PROGRAM:**

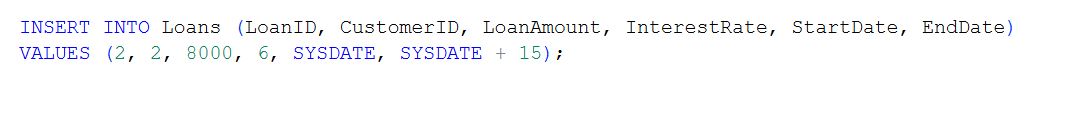


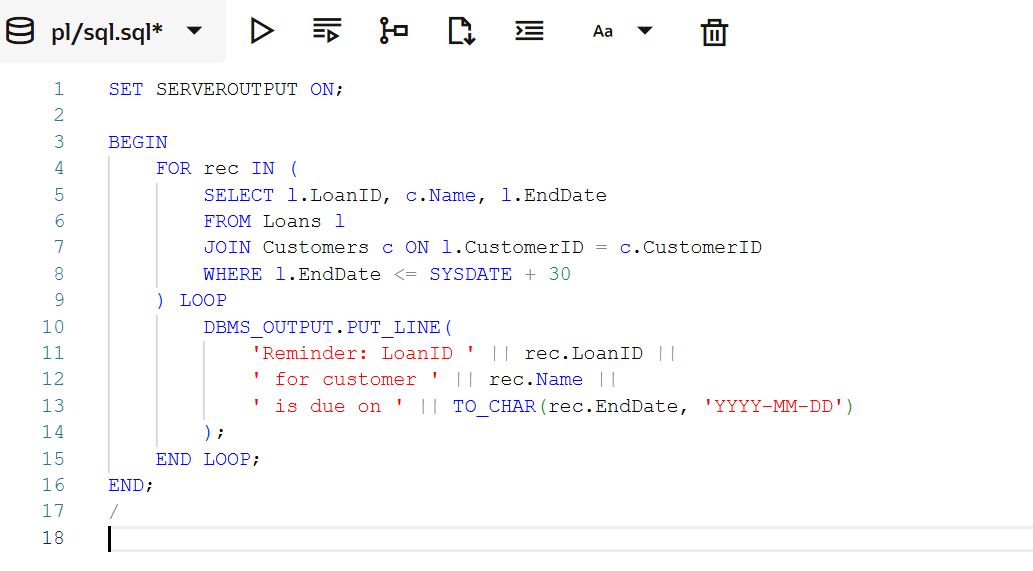
**OUTPUT:**





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

****

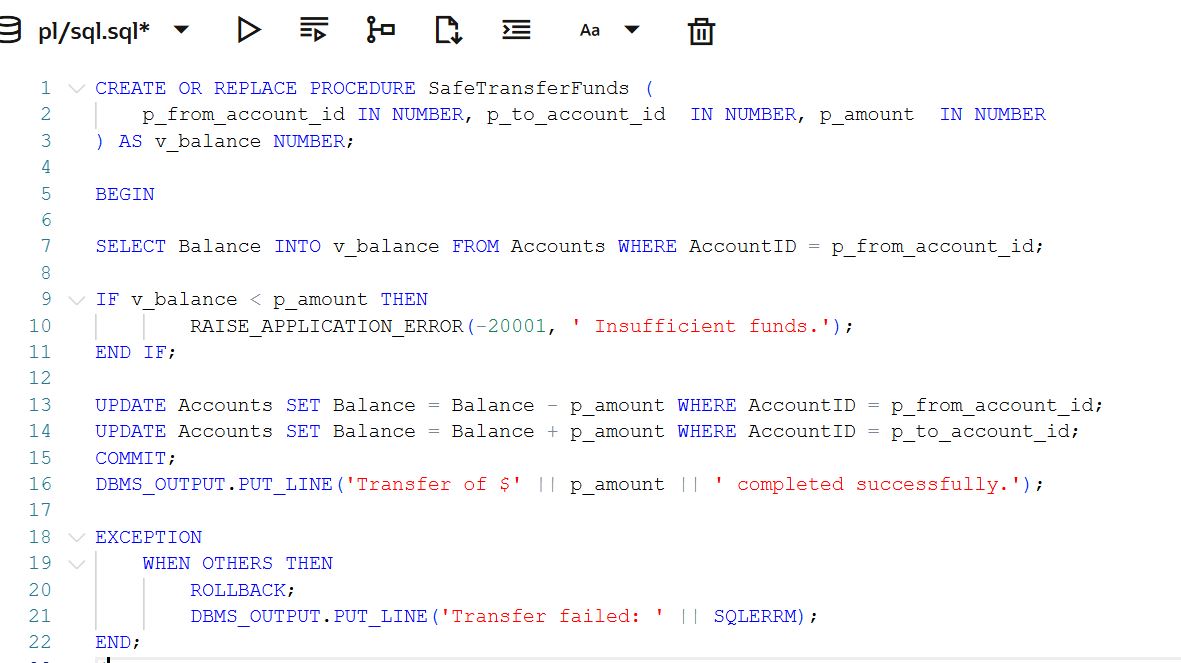
****

**OUTPUT:**

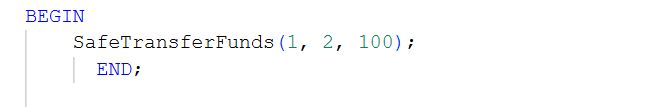
****

**Exercise 2: Error Handling**

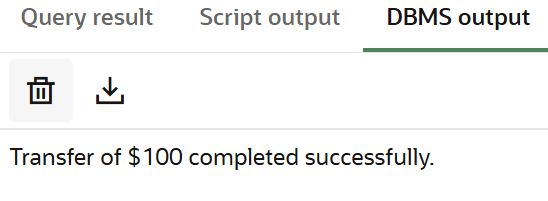
**1) Write a stored procedure SafeTransferFunds that transfers funds between two accounts. Ensure that if any error occurs (e.g., insufficient funds), an appropriate error message is logged and the transaction is rolled back.**

****

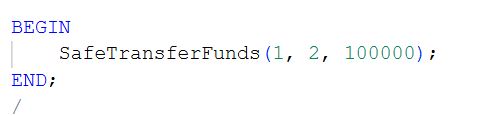
**INPUT 1:**

****

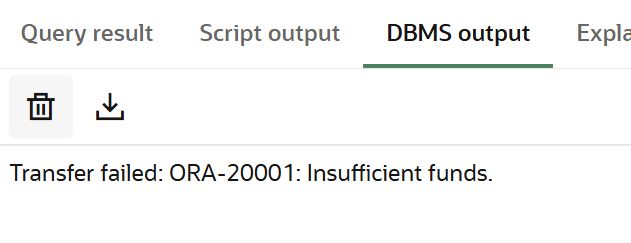
**OUTPUT 1:**

****

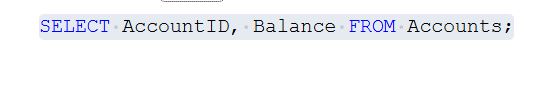
**INPUT 2:**

****

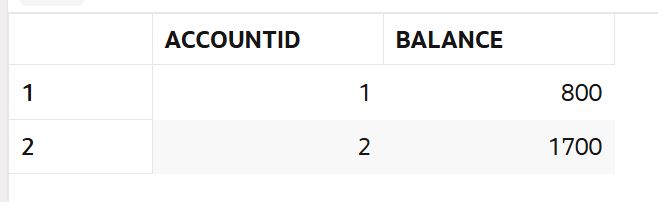
**OUTPUT 2:**

****

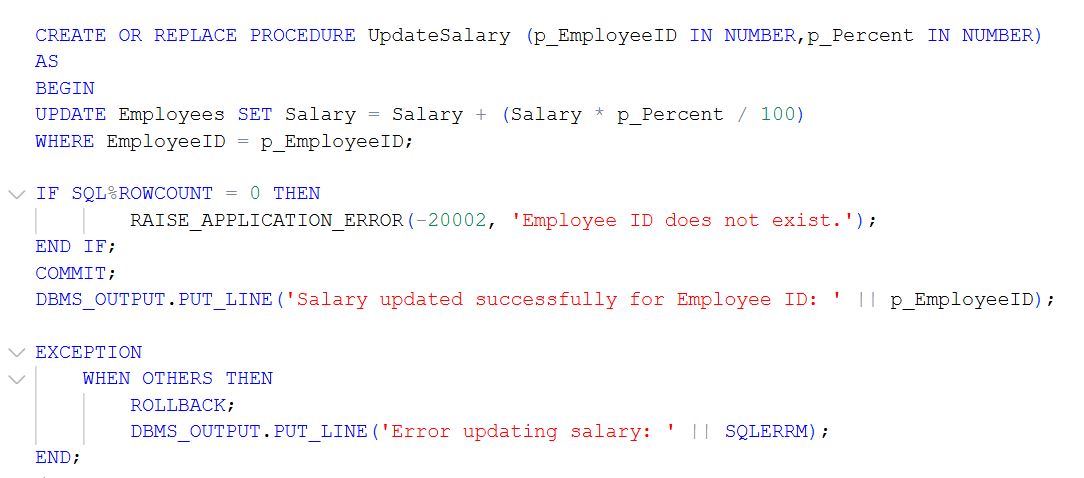
**VERIFY:**

****

**OUTPUT TABLE :**

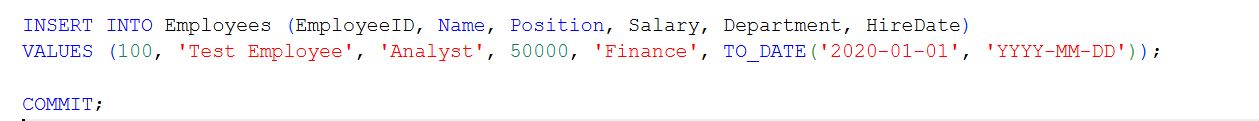
****

**2)** **Write a stored procedure UpdateSalary that increases the salary of an employee by a given percentage. If the employee ID does not exist, handle the exception and log an error message.**

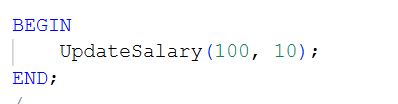
****

****

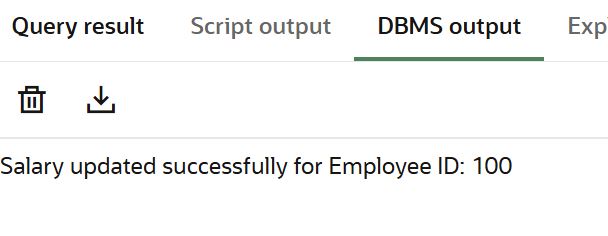
Insert into table employee :

****

Verify:

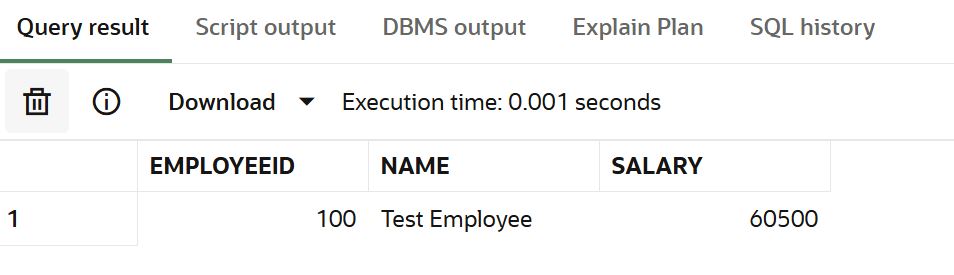
****

Output:

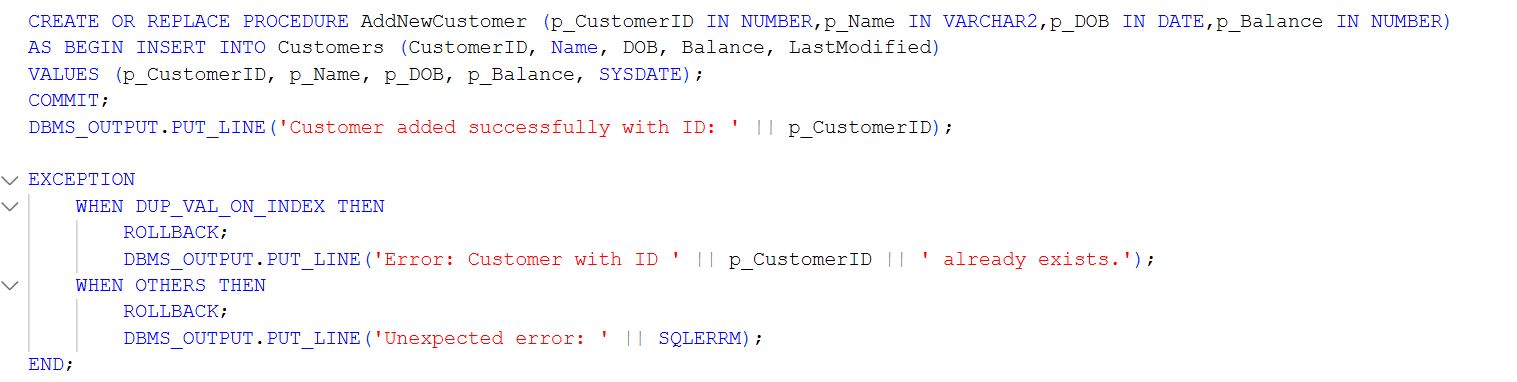
****

****

**OUTPUT:**

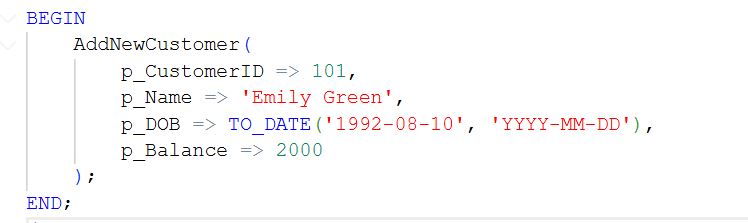
****

**3)** **Write a stored procedure AddNewCustomer that inserts a new customer into the Customers table. If a customer with the same ID already exists, handle the exception by logging an error and preventing the insertion.**

****

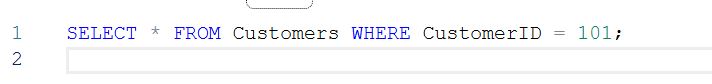
****

**VERIFY:**

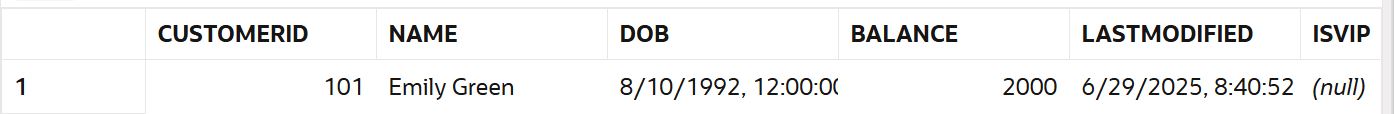
****

**OUTPUT:**

****

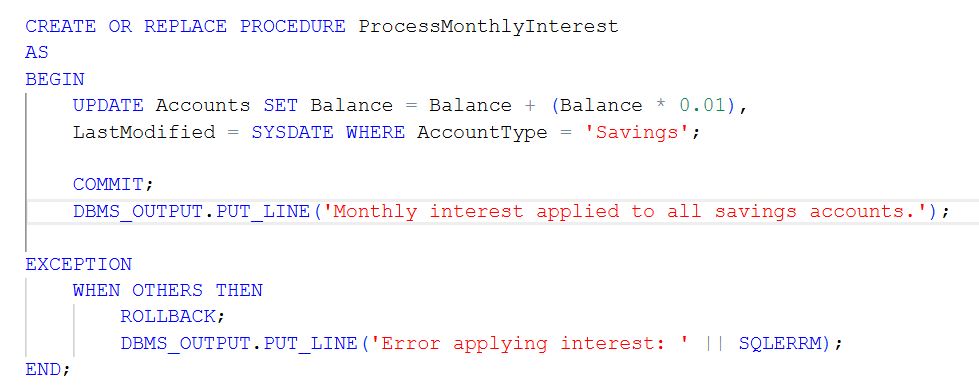
****

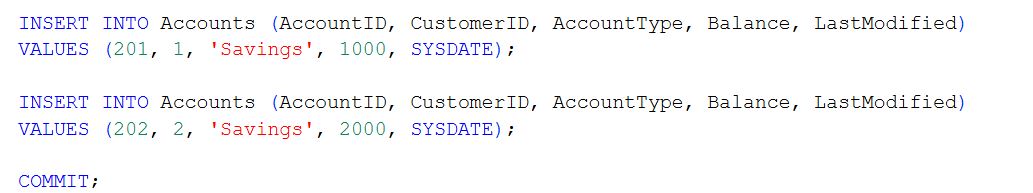
**OUTPUT:**

****

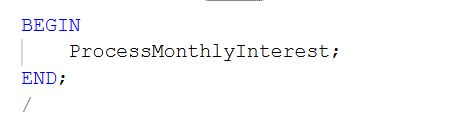
**Exercise 3: Stored Procedures**

**1) 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.**

****

****

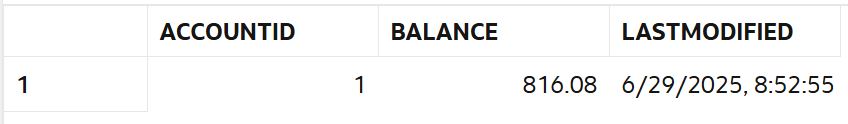
**VERIFY:**

****

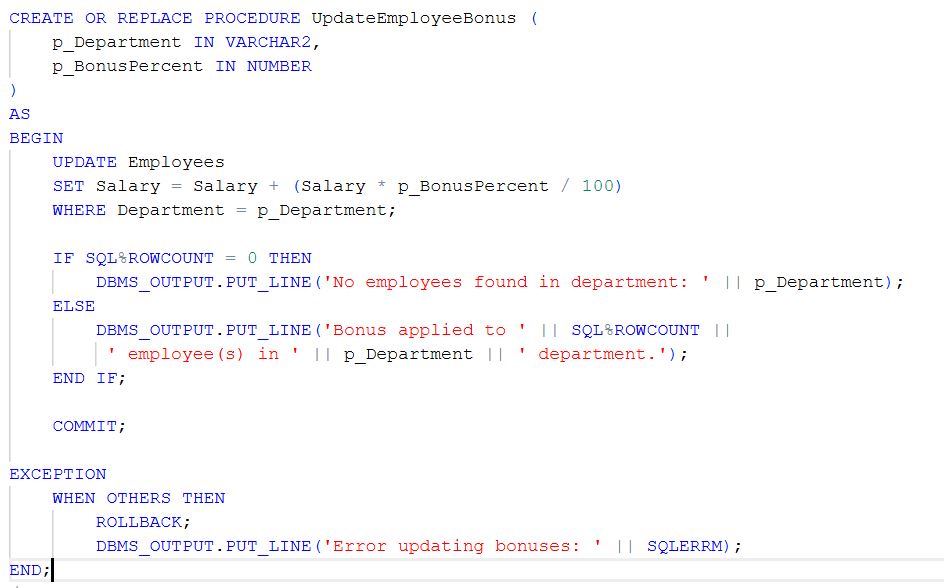
**OUTPUT:**

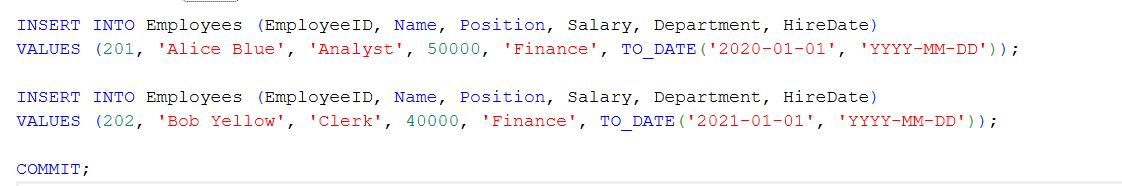
****

**OUTPUT:**

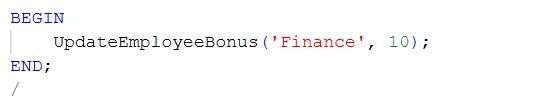
****

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

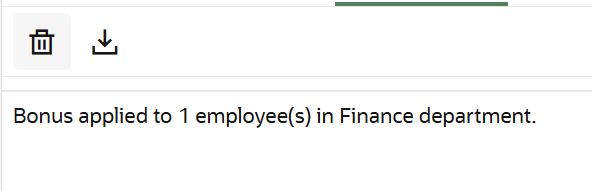
****

****

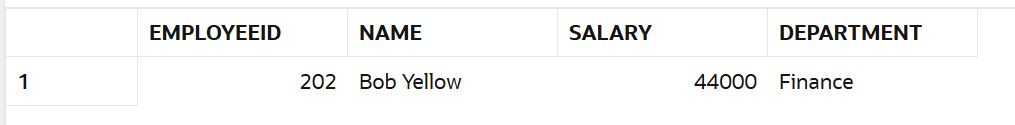
**VERIFY:**

****

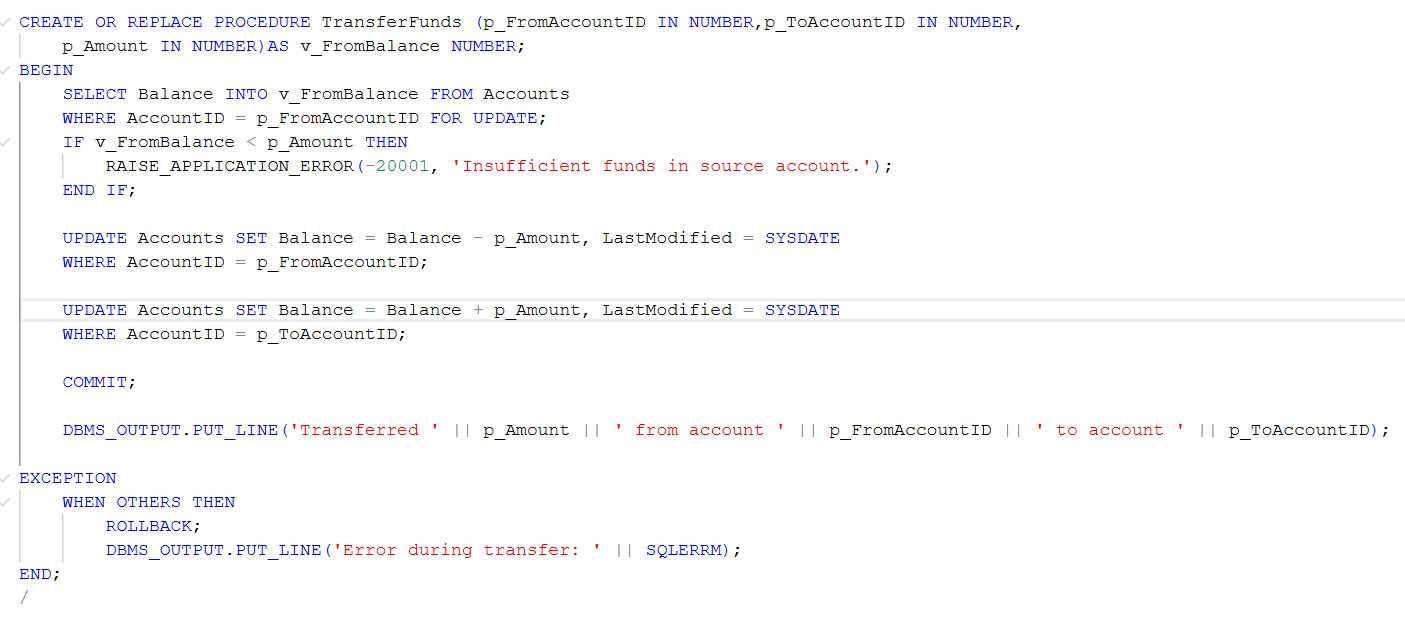
**OUTPUT:**

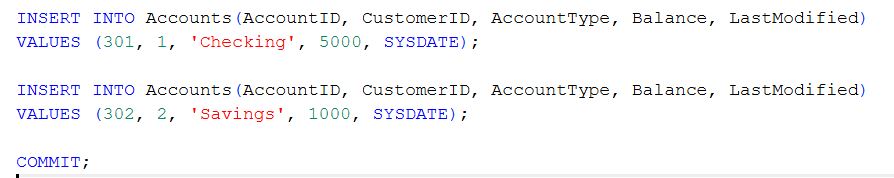
****

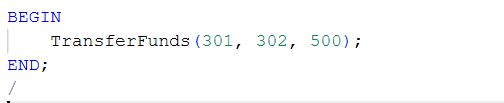
**OUTPUT:**

****

**3)** **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.**

****

****

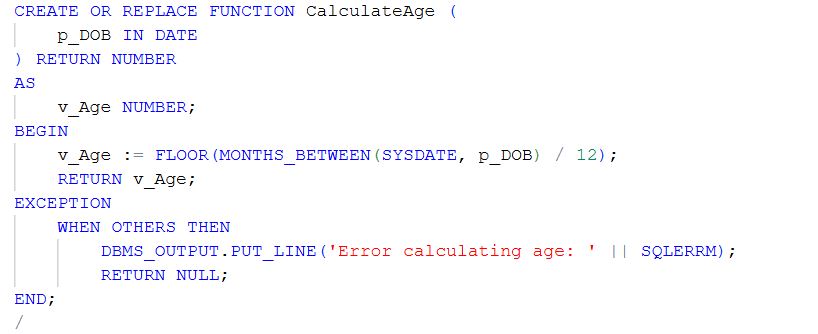
**VERIFY:  
**

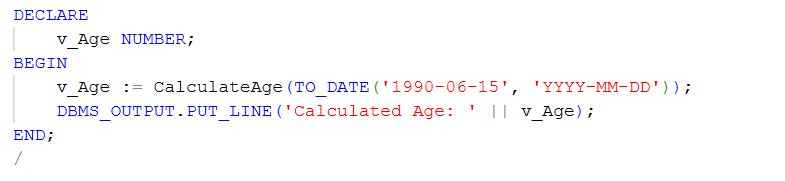
**OUTPUT:**

****

**Exercise 4: Functions**

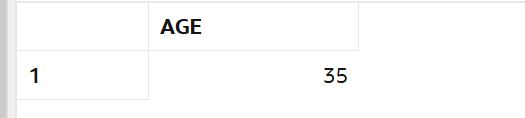
**1) Write a function CalculateAge that takes a customer's date of birth as input and returns their age in years.**

****

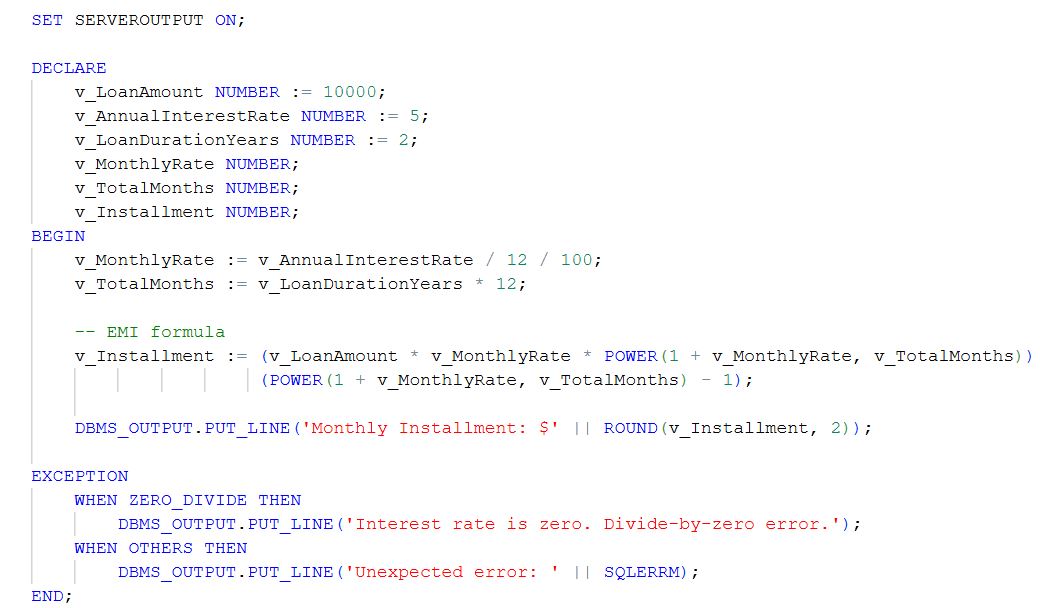
****

**OUTPUT:**

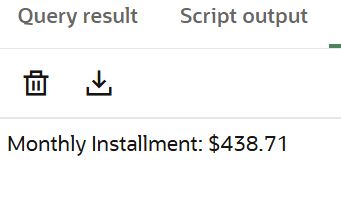
****

****

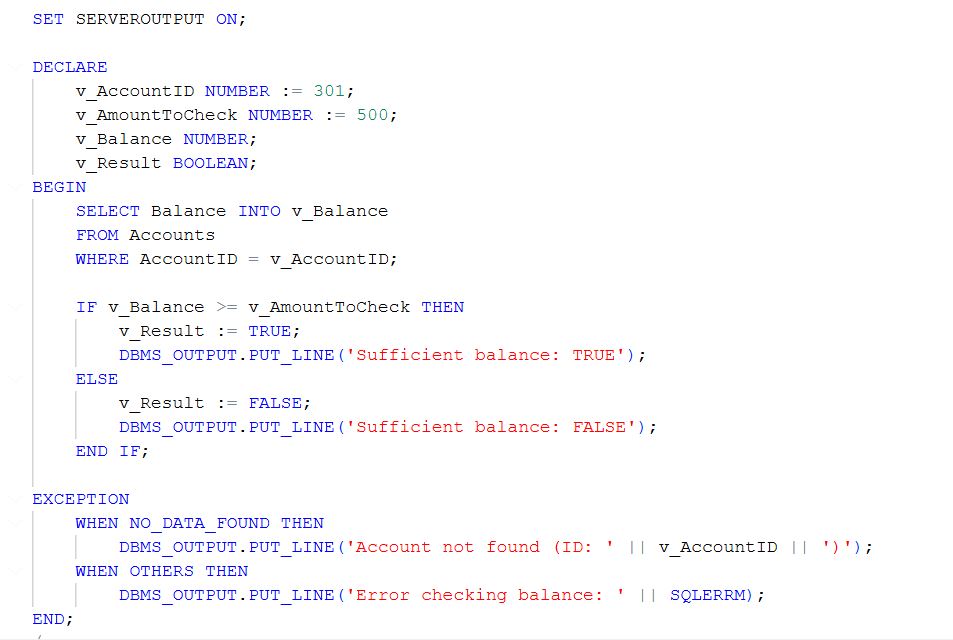
**2)** **Write a function CalculateMonthlyInstallment that takes the loan amount, interest rate, and loan duration in years as input and returns the monthly installment amount.**

****

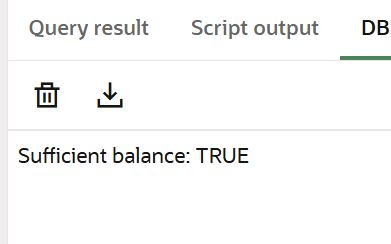
**OUTPUT:**

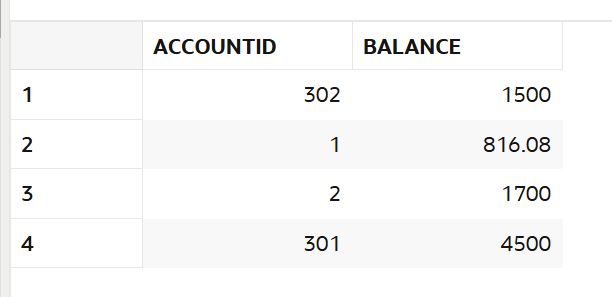
****

**3)** **Write a function HasSufficientBalance that takes an account ID and an amount as input and returns a boolean indicating whether the account has at least the specified amount.**

****

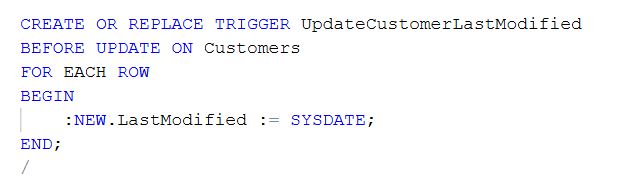
**OUTPUT:**

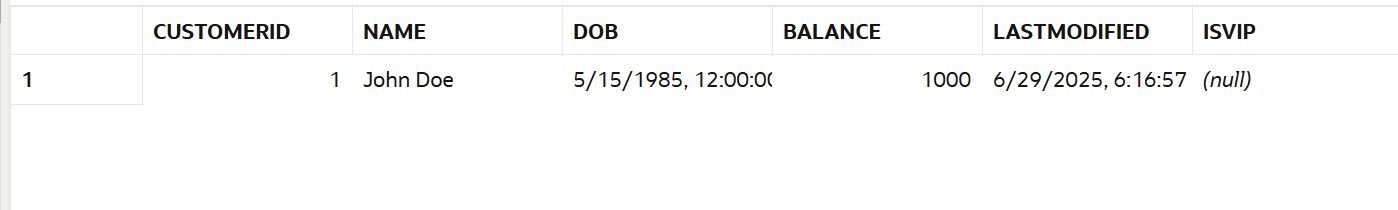
****

****

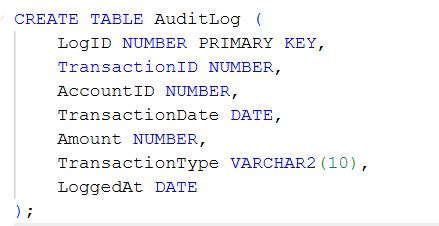
**Exercise 5: Triggers**

**1) Write a trigger UpdateCustomerLastModified that updates the LastModified column of the Customers table to the current date whenever a customer's record is updated.**

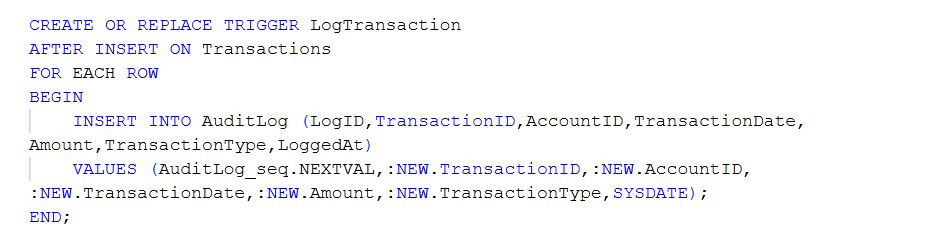
****

****

**2)** **Maintain an audit log for all transactions.**

****

**TRIGGER:**

****

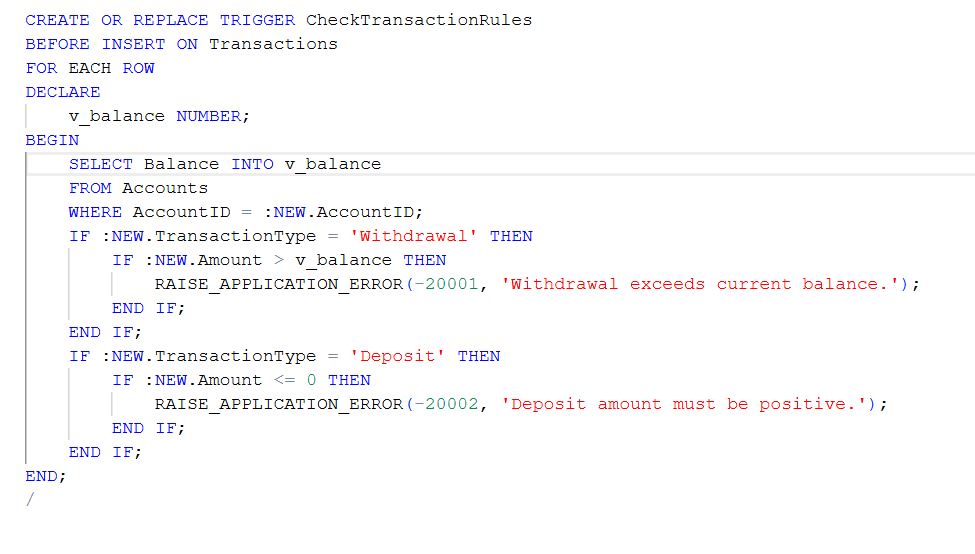
****

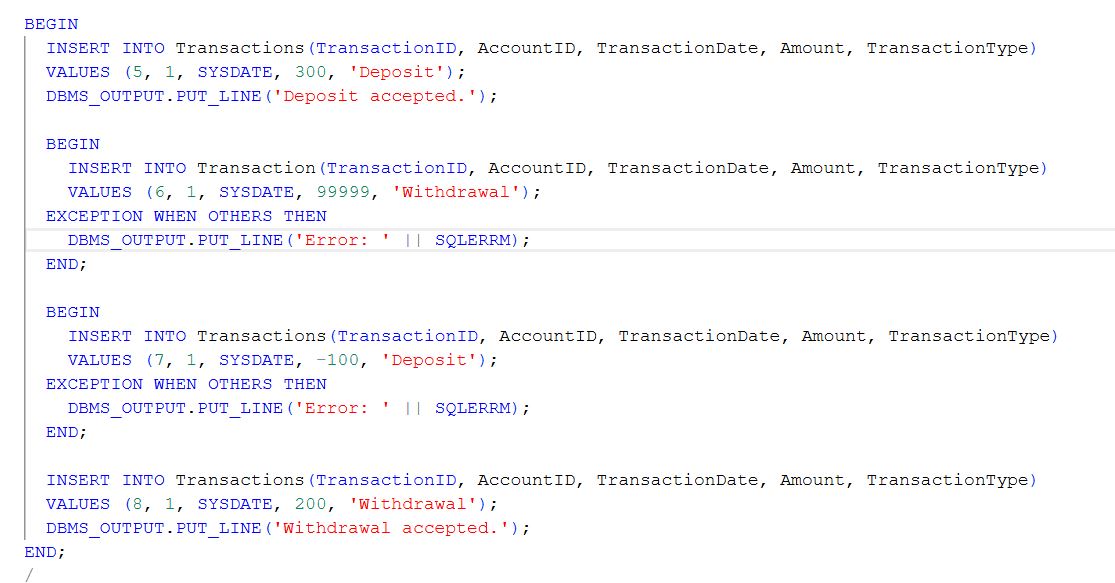
****

**OUTPUT:**

****

**3)** **Enforce business rules on deposits and withdrawals**

****

****

**OUTPUT:**

Deposit accepted.

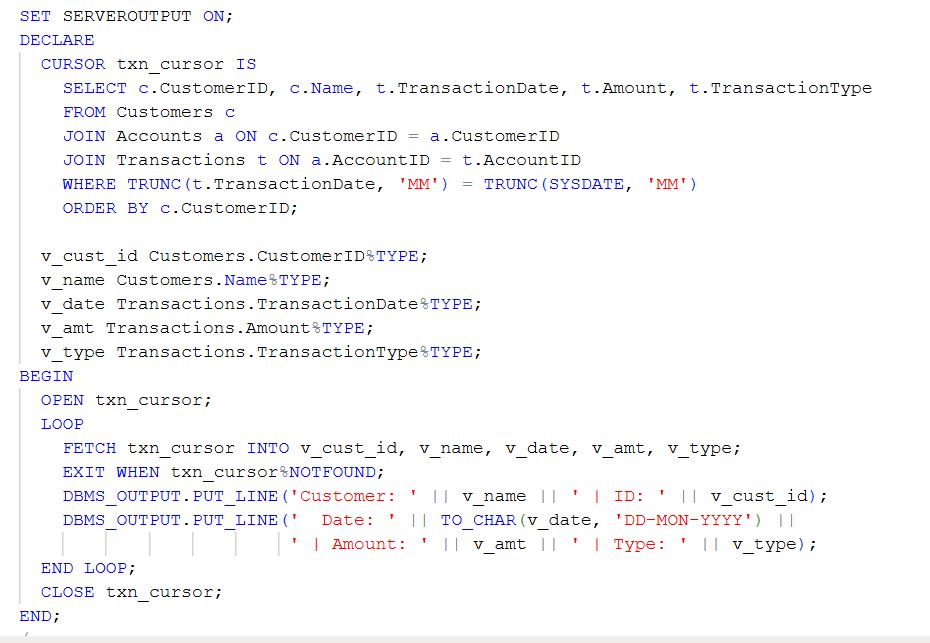
Error: ORA-20001: Withdrawal exceeds current balance.

Error: ORA-20002: Deposit amount must be positive.

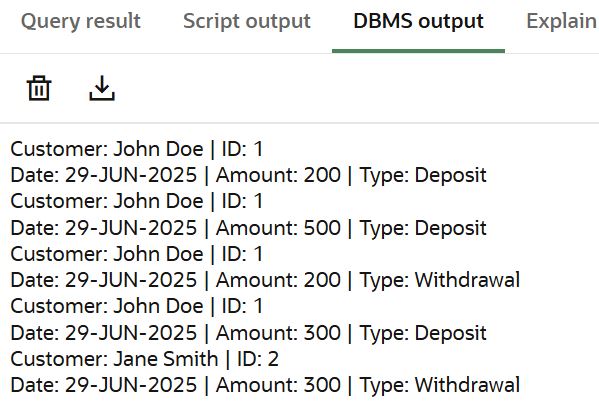
Withdrawal accepted.

**Exercise 6: Cursors**

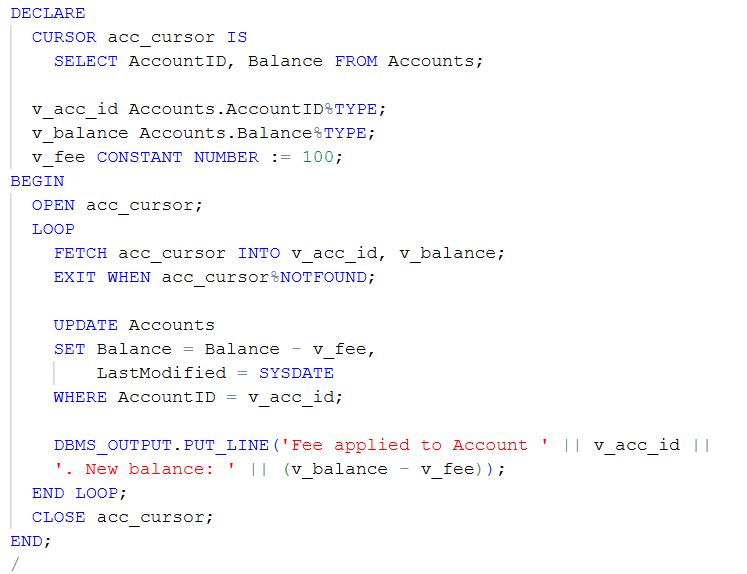
**1)**Generate monthly statements for all customers.



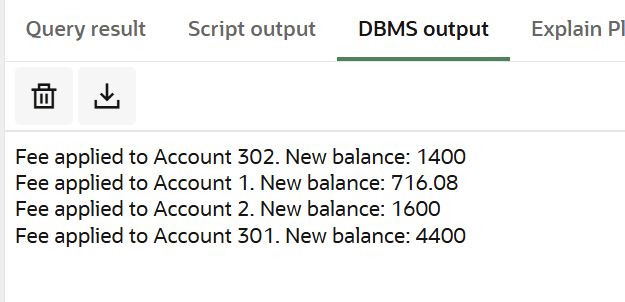
OUTPUT:



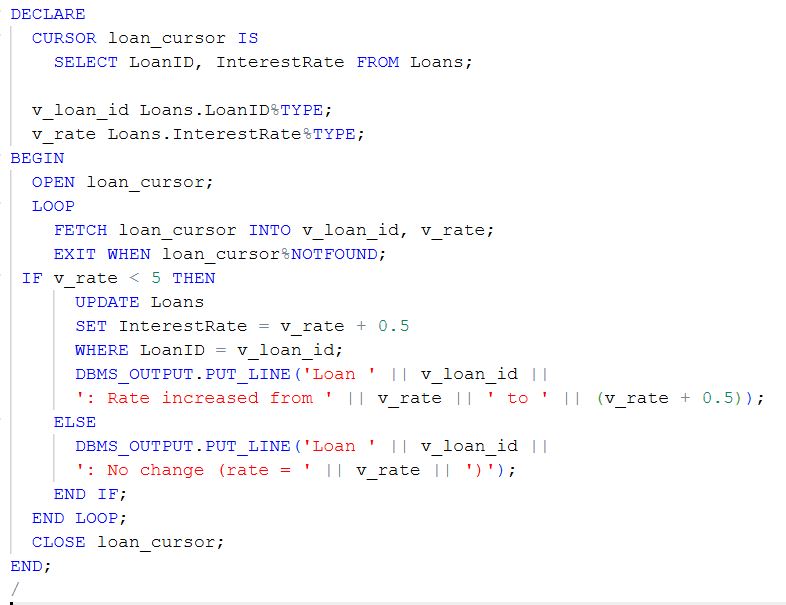
**2:** Apply annual fee to all accounts.



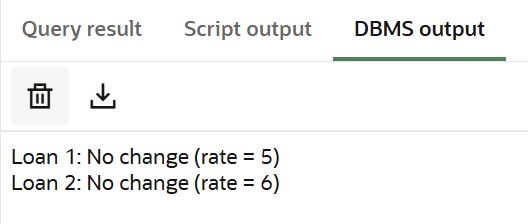
OUTPUT:



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

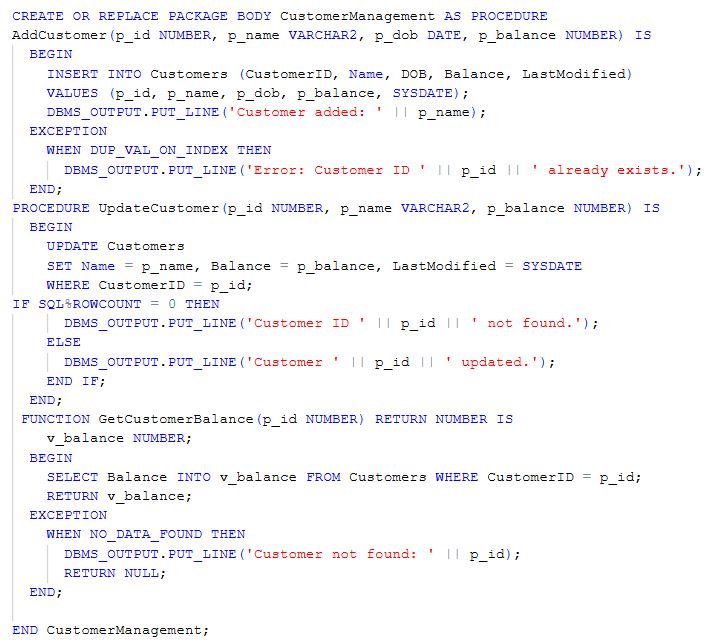


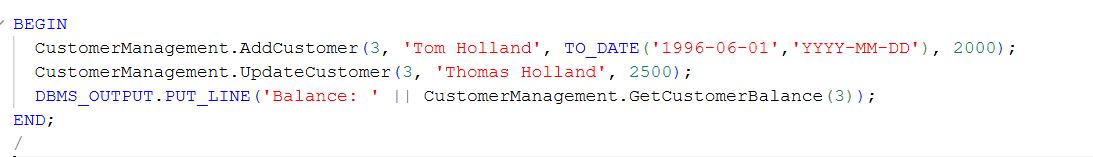
OUTPUT:



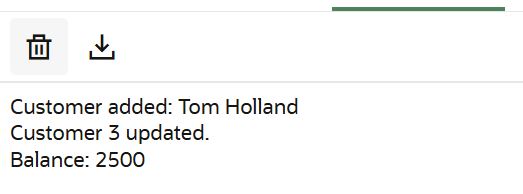
**Exercise 7: Packages**

**1:** Group all customer-related procedures and functions into a package.

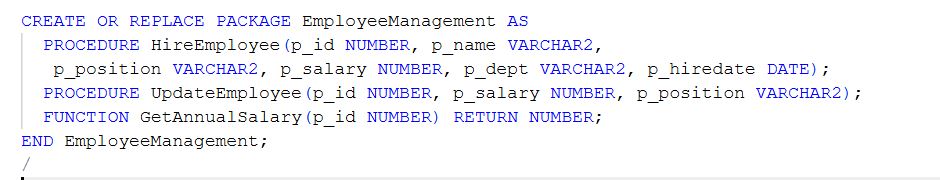


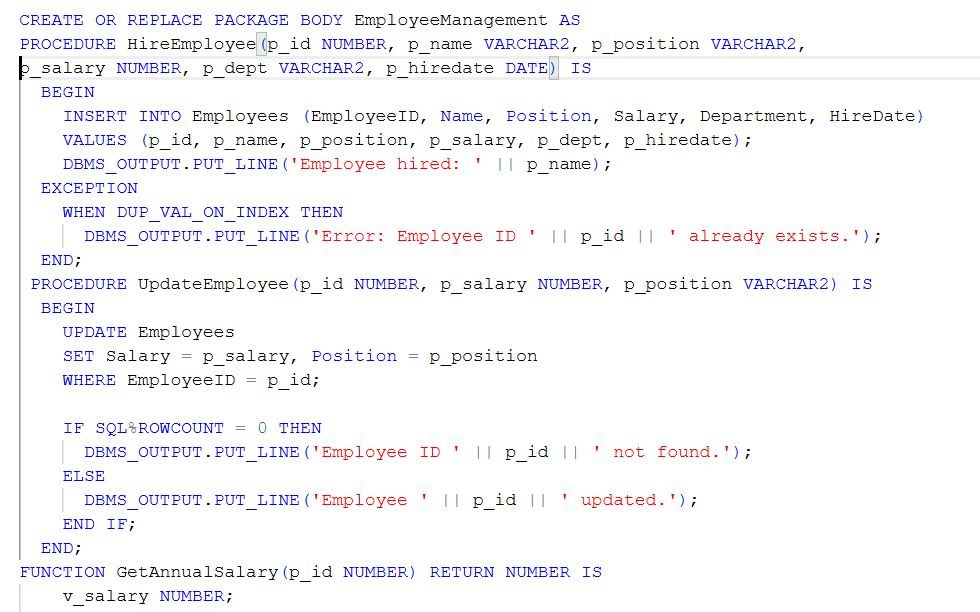


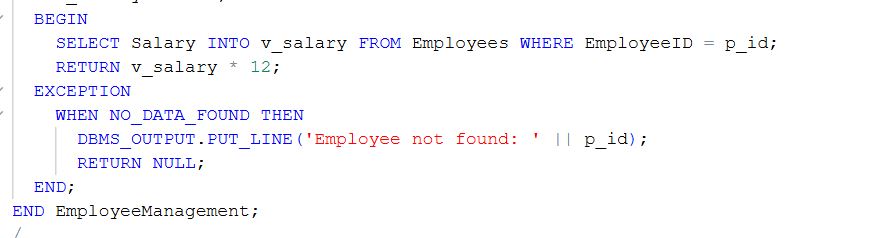
OUTPUT:

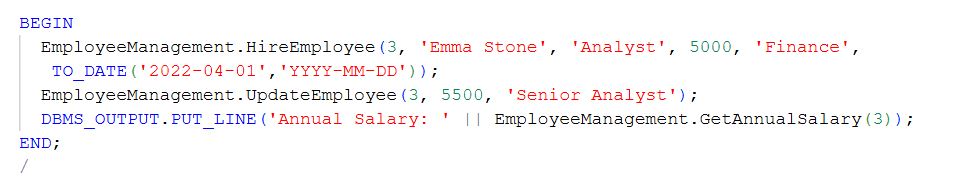


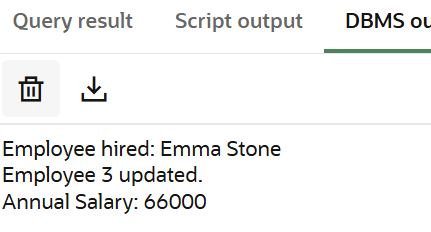
**2:** Create a package to manage employee data.



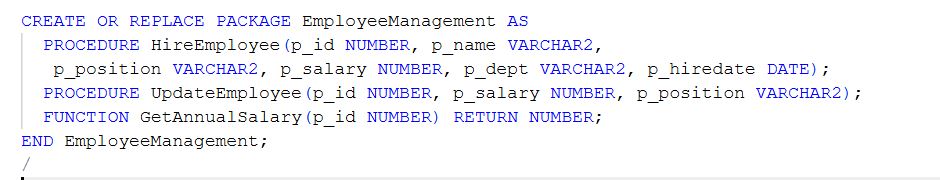


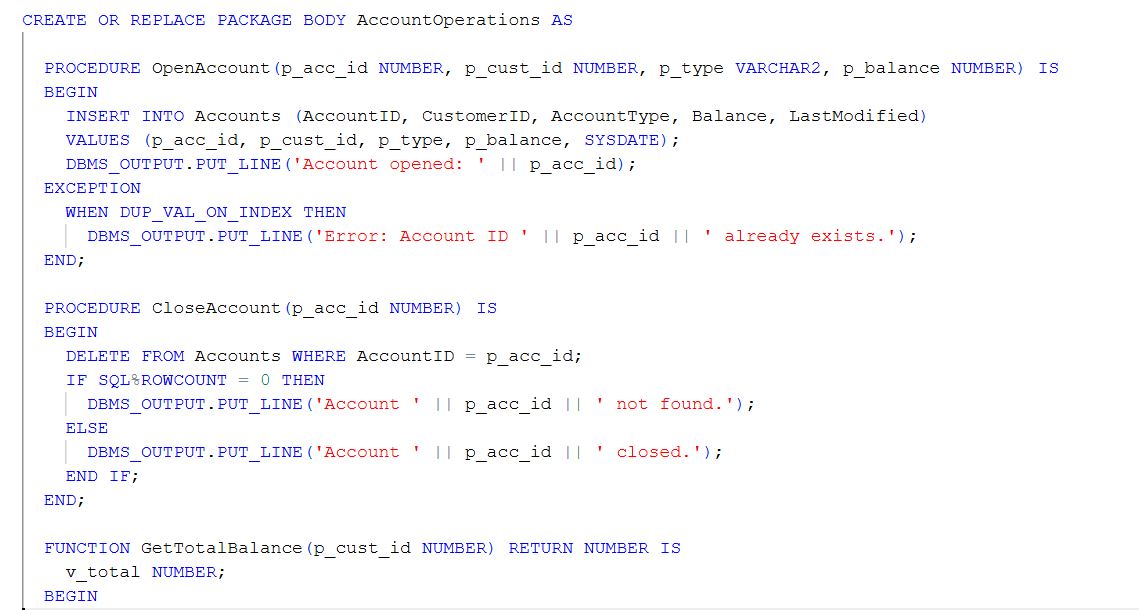


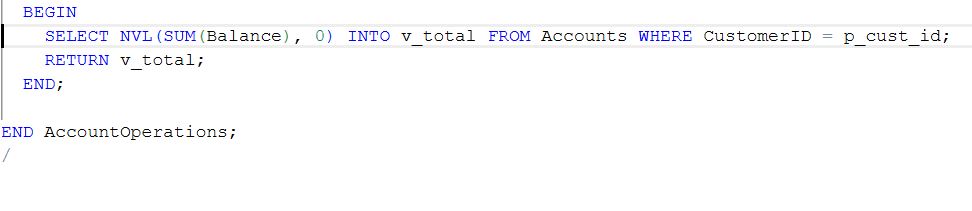


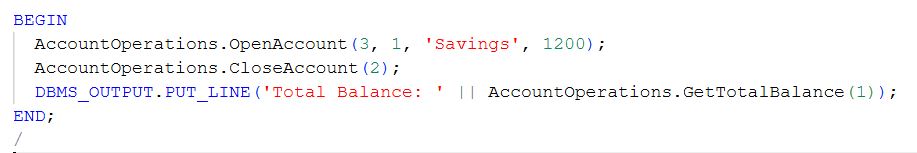
OUTPUT:  


**3:** Group all account-related operations into a package.









OUTPUT:

