After inserting all the values and creating all the tables

Welcome Page | BHAVYA | ACCOUNTS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes

Sort.. | Filter:

	ACCOUNTID	CUSTOMERID	ACCOUNTTYPE	BALANCE	LASTMODIFIED
1	1	1	Savings	1000	25-06-25
2	2	2	Checking	1500	25-06-25
3	3	3	Savings	12000	25-06-25

Welcome Page | BHAVYA | LOANS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	LOANID	CUSTOMERID	LOANAMOUNT	INTERESTRATE	STARTDATE	ENDDATE
1	1	1	5000	5	25-06-25	25-06-30
2	2	3	8000	6	25-06-25	15-07-25

Welcome Page | BHAVYA | EMPLOYEES

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	EMPLOYEEID	NAME	POSITION	SALARY	DEPARTMENT	HIREDATE
1	1	Alice Johnson	Manager	70000	HR	15-06-15
2	2	Bob Brown	Developer	60000	IT	20-03-17
3	3	Test Engineer	Tester	50000	QA	25-06-25

Welcome Page | BHAVYA | TRANSACTIONS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	TRANSACTIONID	ACCOUNTID	TRANSACTIONDATE	AMOUNT	TRANSACTIONTYPE
1	1	1	25-06-25	200	Deposit
2	2	2	25-06-25	300	Withdrawal

Welcome Page | BHAVYA | CUSTOMERS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	CUSTOMERID	NAME	DOB	BALANCE	LASTMODIFIED	ISVIP
1	1	John Doe	15-05-85	1000	25-06-25	(null)
2	2	Jane Smith	20-07-90	1500	25-06-25	(null)
3	3	Senior Citizen	01-01-50	12000	25-06-25	(null)

Exercise 1: Control Structures

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

- **Question:** 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:

```
SET SERVEROUTPUT ON;


BEGIN

  FOR cust IN (

    SELECT c.CustomerID, l.LoanID, l.InterestRate

    FROM Customers c
```

```
    JOIN Loans l ON c.CustomerID = l.CustomerID
    WHERE MONTHS_BETWEEN(SYSDATE, c.DOB)/12 > 60
  ) LOOP
    UPDATE Loans
    SET InterestRate = InterestRate - 1
    WHERE LoanID = cust.LoanID;

    DBMS_OUTPUT.PUT_LINE('1% discount applied to CustomerID ' || cust.CustomerID);
  END LOOP;
  COMMIT;
END;
/
```

OUTPUT:

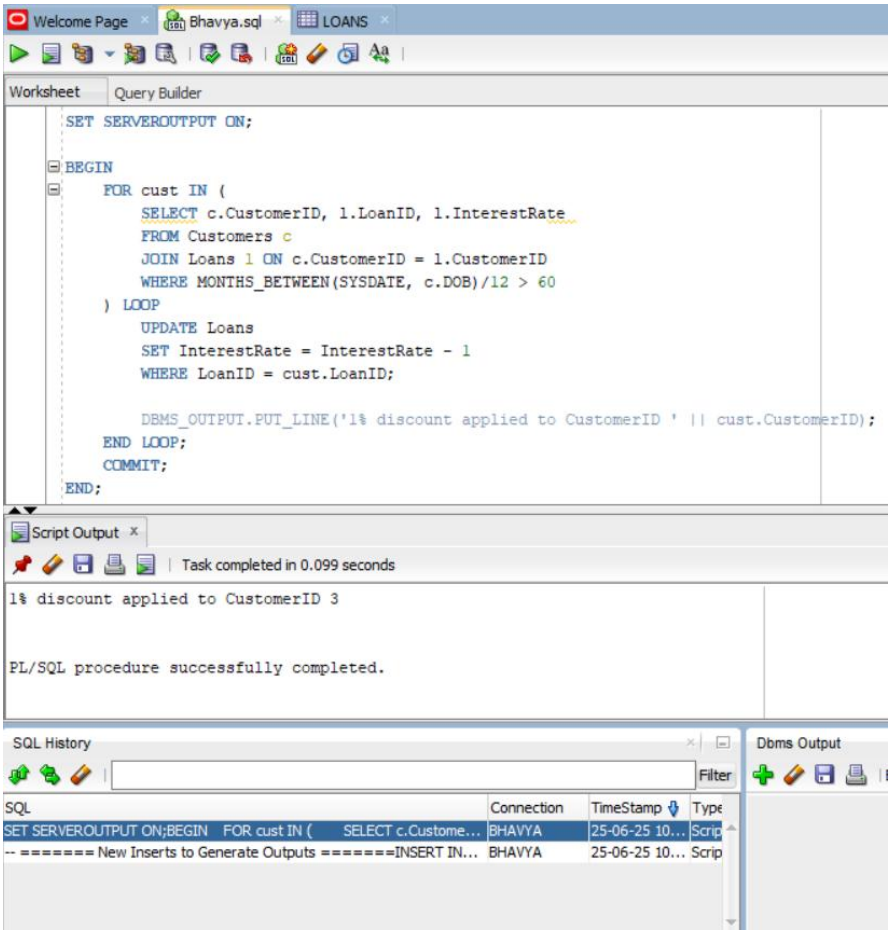
Welcome Page | Bhavya.sql | LOANS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions

Sort.. | Filter:

	LOANID	CUSTOMERID	LOANAMOUNT	INTERESTRATE	STARTDATE	ENDDATE
1	1	1	5000	5	25-06-25	25-06-30
2	2	3	8000	5	25-06-25	15-07-25

Before the intrest rate was 6 now it became 5



Exercise 1: Control Structures

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

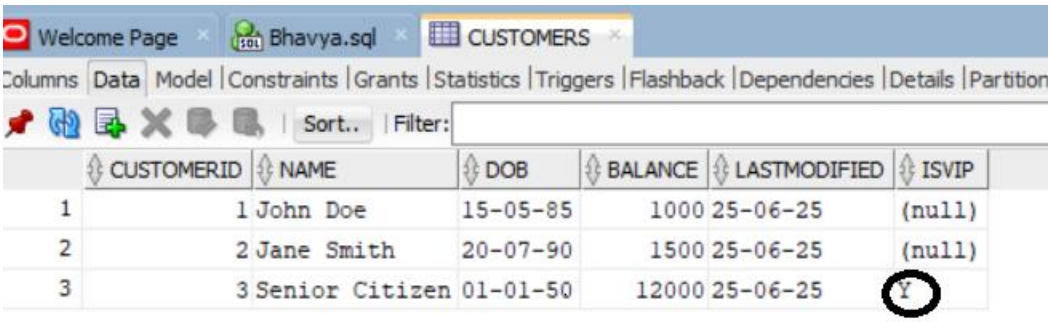
Question: Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.

CODE:

```
BEGIN
  FOR cust IN (
    SELECT CustomerID, Balance FROM Customers
  ) LOOP
    IF cust.Balance > 10000 THEN
      UPDATE Customers
```

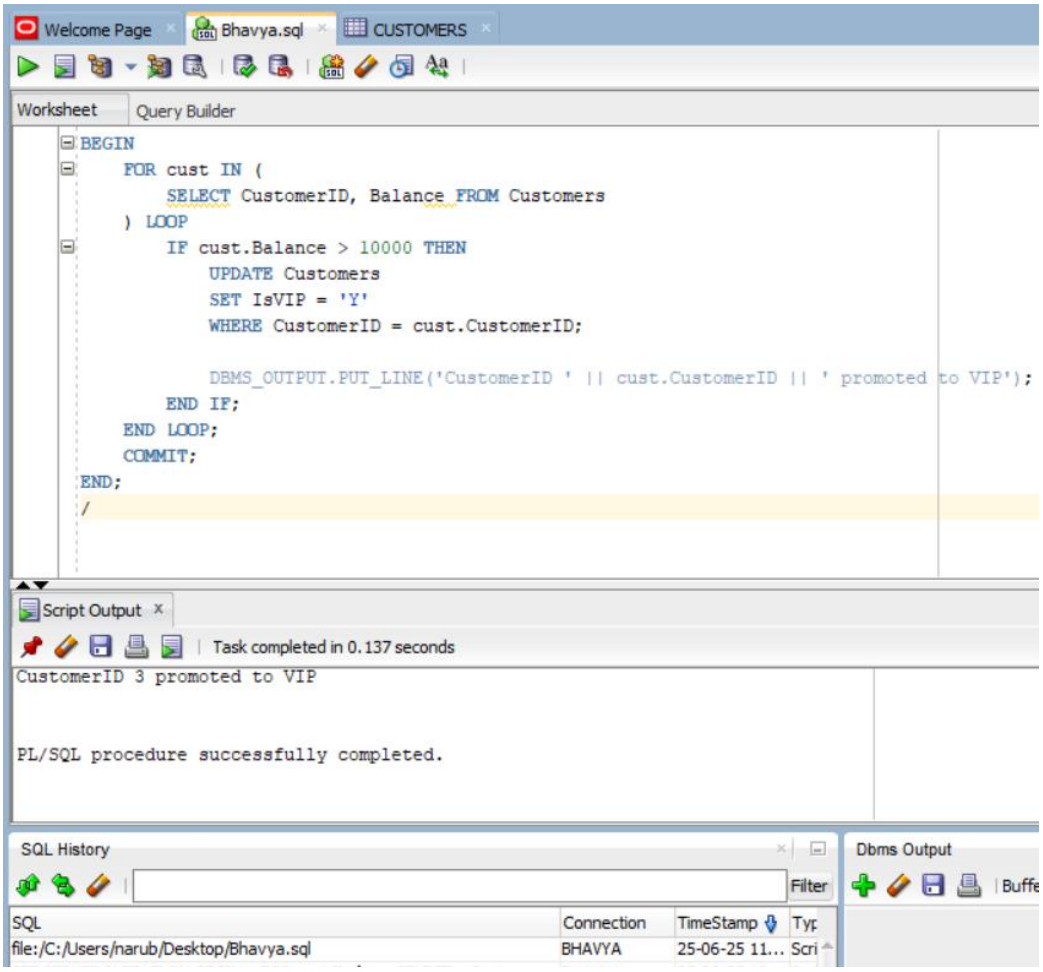
```
        SET IsVIP = 'Y'
        WHERE CustomerID = cust.CustomerID;

        DBMS_OUTPUT.PUT_LINE('CustomerID ' || cust.CustomerID || ' promoted to VIP');
      END IF;
    END LOOP;
    COMMIT;
END;
/
```

OUTPUT:



	CUSTOMERID	NAME	DOB	BALANCE	LASTMODIFIED	ISVIP
1	1	John Doe	15-05-85	1000	25-06-25	(null)
2	2	Jane Smith	20-07-90	1500	25-06-25	(null)
3	3	Senior Citizen	01-01-50	12000	25-06-25	Y

ISVIP of the customer3 has changed from null to Y which indicate True as he have balance>10000



Exercise 1: Control Structures

Scenario 3: The bank wants to send reminders to customers whose loans are due within the next 30 days.

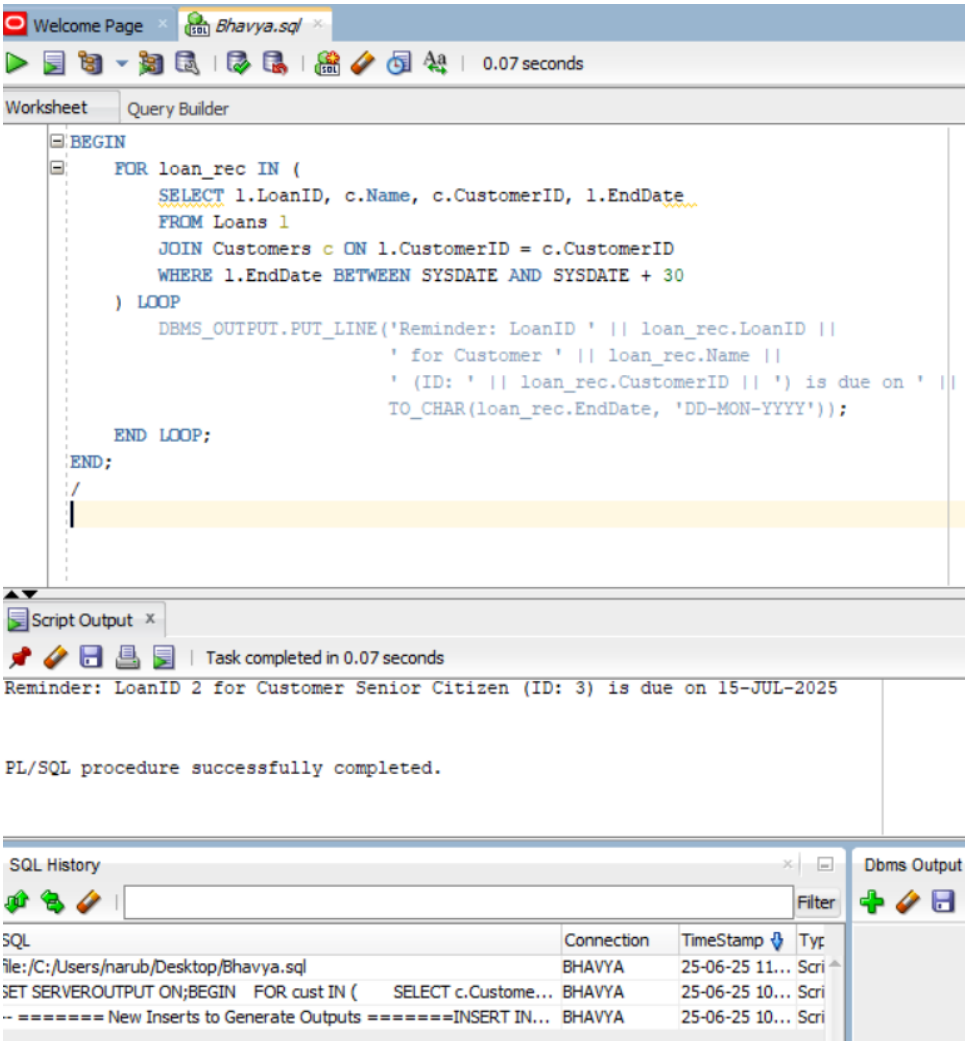
Question: Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

CODE:

```
BEGIN
  FOR loan_rec IN (
    SELECT l.LoanID, c.Name, c.CustomerID, l.EndDate
    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: LoanID ' || loan_rec.LoanID ||
                ' for Customer ' || loan_rec.Name ||
                ' (ID: ' || loan_rec.CustomerID || ') is due on ' ||
                TO_CHAR(loan_rec.EndDate, 'DD-MON-YYYY'));
```

```
    END LOOP;
END;
/
```

OUTPUT:



Exercise 3: Stored Procedures

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

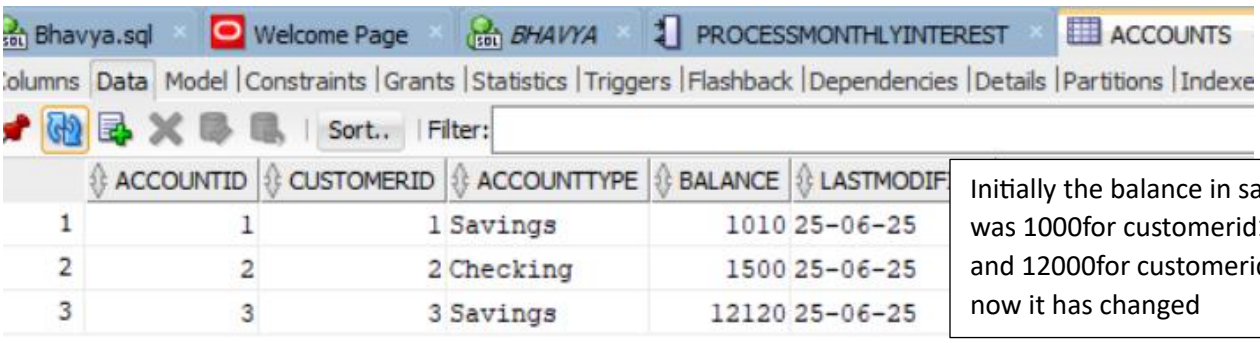
- **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

CODE:

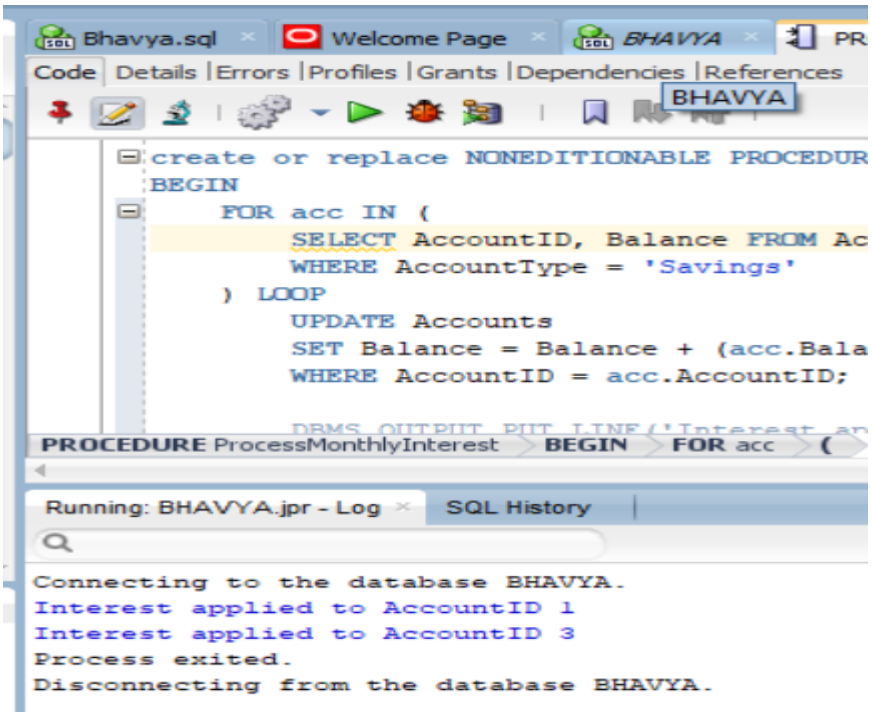
```
CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS
BEGIN
    FOR acc IN (
```

```
      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('Interest applied to AccountID ' || acc.AccountID);
   END LOOP;
   COMMIT;
END;
/
BEGIN
   ProcessMonthlyInterest;
END;
```

OUTPUT:



Initially the balance in saving was 1000for customerid1 and 12000for customerid3 now it has changed



Exercise 3: Stored Procedures

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:

```
SET SERVEROUTPUT ON;
CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(
   p_Department IN VARCHAR2,
   p_BonusPercent IN NUMBER
) IS
BEGIN
   FOR emp IN (
      SELECT EmployeeID, Salary FROM Employees
      WHERE Department = p_Department
   ) LOOP
      UPDATE Employees
      SET Salary = Salary + (emp.Salary * p_BonusPercent / 100)
      WHERE EmployeeID = emp.EmployeeID;

      DBMS_OUTPUT.PUT_LINE('Bonus applied to EmployeeID ' || emp.EmployeeID);
   END LOOP;
   COMMIT;
END;
/
--CallingTHe procedure
BEGIN
   UpdateEmployeeBonus('IT', 5);
END;
```

OUTPUT:

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Par

Sort.. | Filter:

	EMPLOYEEID	NAME	POSITION	SALARY	DEPARTMENT	HIREDATE
1	1	Alice Johnson	Manager	70000	HR	15-06-15
2	2	Bob Brown	Developer	63000	IT	20-03-17
3	3	Test Engineer	Tester	50000	QA	25-06-25

BHAVYA | EMPLOYEES | UPDATEEMPLOYEEBONUS

Code | Details | Errors | Profiles | Grants | Dependencies | References

```
create or replace NONEDITIONABLE PROCEDURE U
    p_Department IN VARCHAR2,
    p_BonusPercent IN NUMBER
) IS
BEGIN
    FOR emp IN
```

PROCEDURE UpdateEmployeeBonus > BEGIN > FOR emp > (

Running: BHAVYA.jpr - Log | SQL History

```
Connecting to the database BHAVYA.
Bonus applied to EmployeeID 2
Process exited.
Disconnecting from the database BHAVYA.
```

Exercise 3: Stored Procedures

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: SET SERVEROUTPUT ON;

```
CREATE OR REPLACE PROCEDURE TransferFunds(
   p_FromAccountID IN NUMBER,
   p_ToAccountID IN NUMBER,
   p_Amount IN NUMBER
) IS
   v_FromBalance NUMBER;
```

```
BEGIN
   -- Check available balance
   SELECT Balance INTO v_FromBalance
   FROM Accounts
   WHERE AccountID = p_FromAccountID;

   IF v_FromBalance < p_Amount THEN
      RAISE_APPLICATION_ERROR(-20001, 'Insufficient funds.');
   END IF;

   -- Perform transfer
   UPDATE Accounts
   SET Balance = Balance - p_Amount
   WHERE AccountID = p_FromAccountID;

   UPDATE Accounts
   SET Balance = Balance + p_Amount
   WHERE AccountID = p_ToAccountID;

   COMMIT;
   DBMS_OUTPUT.PUT_LINE('Transfer successful.');
EXCEPTION
   WHEN OTHERS THEN
      ROLLBACK;
      DBMS_OUTPUT.PUT_LINE('Transfer failed: ' || SQLERRM);
END;
/
BEGIN
   TransferFunds(3, 2, 500);  -- $500 from Account 3 to Account 2
END;
/
```

OUTPUT:

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Det

Sort.. | Filter:

	ACCOUNTID	CUSTOMERID	ACCOUNTTYPE	BALANCE	LASTMODIFIED
1	1	1	Savings	1010	25-06-25
2	2	2	Checking	2000	25-06-25
3	3	3	Savings	11620	25-06-25

Bhavya.sql | TRANSFERFUNDS | ACCOUNTS

Code | Errors | Profiles | Grants | References | Dependencies | Details

```
create or replace NONEDITIONABLE PROCEDUR
    p_FromAccountID IN NUMBER,
    p_ToAccountID IN NUMBER,
    p_Amount IN NUMBER
) IS
    v_FromBalance NUMBER;
BEGIN
    -- Check available balance
```

Running: BHAVYA.jpr - Log | SQL History

```
Connecting to the database BHAVYA.
Transfer successful.
Process exited.
Disconnecting from the database BHAVYA.
```