**PL\_SQL**

EXERCISE1: CONTROL STRUCTURES

SET SERVEROUTPUT ON;

CREATE TABLE customers (

customer\_id NUMBER PRIMARY KEY,

customer\_name VARCHAR2(100),

age NUMBER,

balance NUMBER,

loan\_interest\_rate NUMBER,

isvip VARCHAR2(5)

);

CREATE TABLE loans (

loan\_id NUMBER PRIMARY KEY,

customer\_id NUMBER REFERENCES customers(customer\_id),

due\_date DATE

);

Insert sample data

INSERT INTO customers VALUES (1, 'John Doe', 65, 12000, 8.5, 'FALSE');

INSERT INTO customers VALUES (2, 'Jane Smith', 45, 9000, 9.0, 'FALSE');

INSERT INTO customers VALUES (3, 'Mark Lee', 70, 10500, 7.5, 'FALSE');

INSERT INTO loans VALUES (101, 1, SYSDATE + 10); -- due soon

INSERT INTO loans VALUES (102, 2, SYSDATE + 45); -- not due

INSERT INTO loans VALUES (103, 3, SYSDATE + 5); -- due soon

COMMIT;

Scenario 1 - Apply 1% discount to interest rate for age > 60

BEGIN

FOR customer\_rec IN (

SELECT customer\_id, age, loan\_interest\_rate FROM customers WHERE age > 60

) LOOP

UPDATE customers

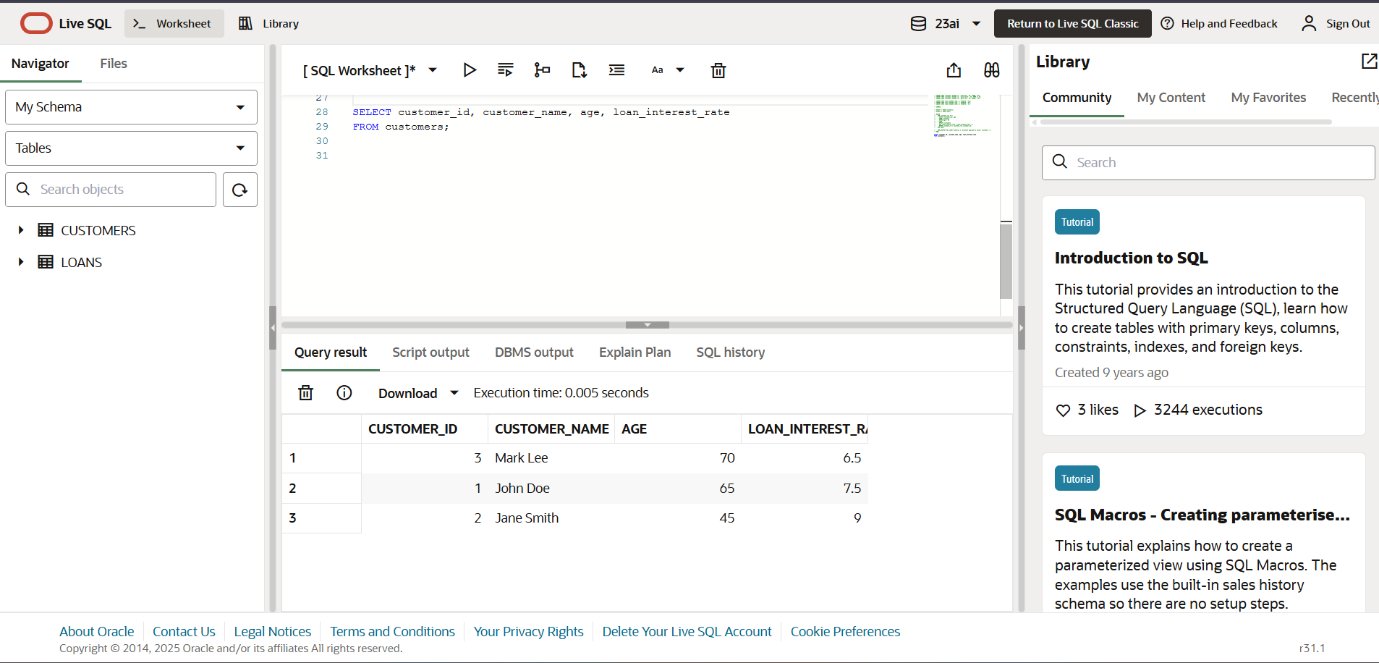
SET loan\_interest\_rate = loan\_interest\_rate - 1

WHERE customer\_id = customer\_rec.customer\_id;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Scenario 1: Interest rate discount applied for customers above 60.');

END;



Scenario 2 - Set IsVIP = TRUE for balance > 10000

BEGIN

FOR customer\_rec IN (

SELECT customer\_id, balance FROM customers WHERE balance > 10000

) LOOP

UPDATE customers

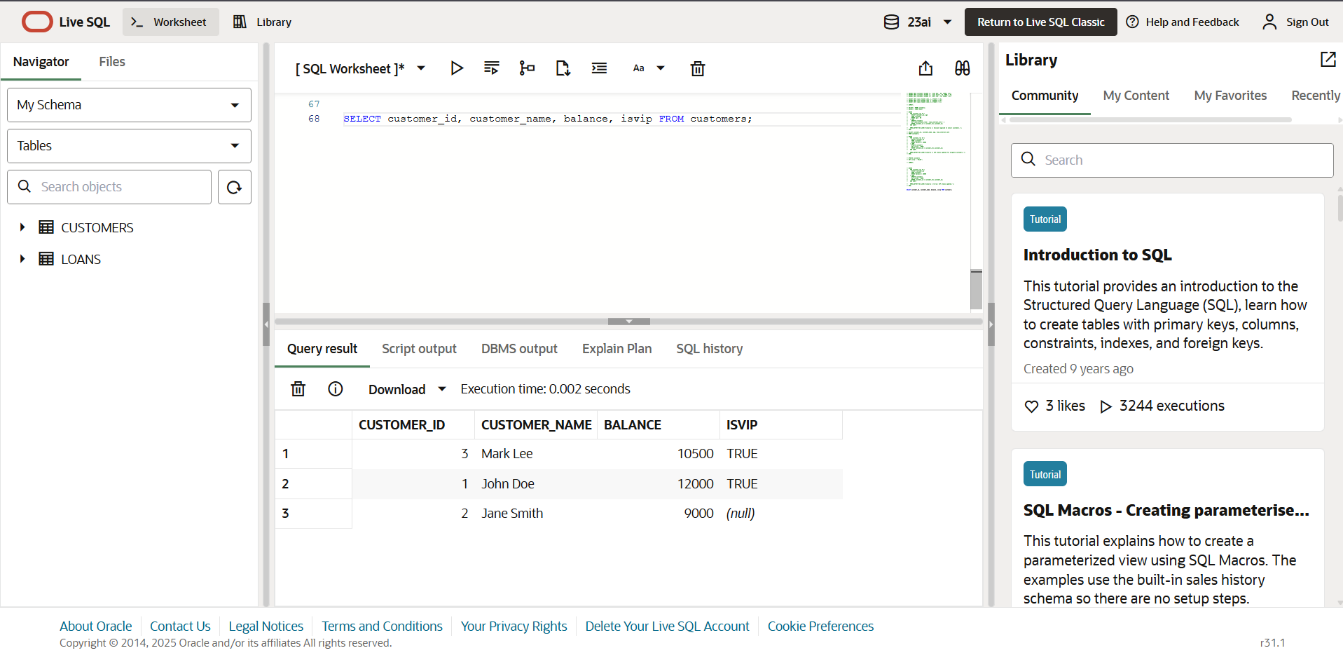
SET isvip = 'TRUE'

WHERE customer\_id = customer\_rec.customer\_id;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Scenario 2: VIP status updated for eligible customers.');

END;



Scenario 3 - Remind customers of loans due in next 30 days

BEGIN

FOR loan\_rec IN (

SELECT l.loan\_id, c.customer\_name, l.due\_date

FROM loans l

JOIN customers c ON l.customer\_id = c.customer\_id

WHERE l.due\_date BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

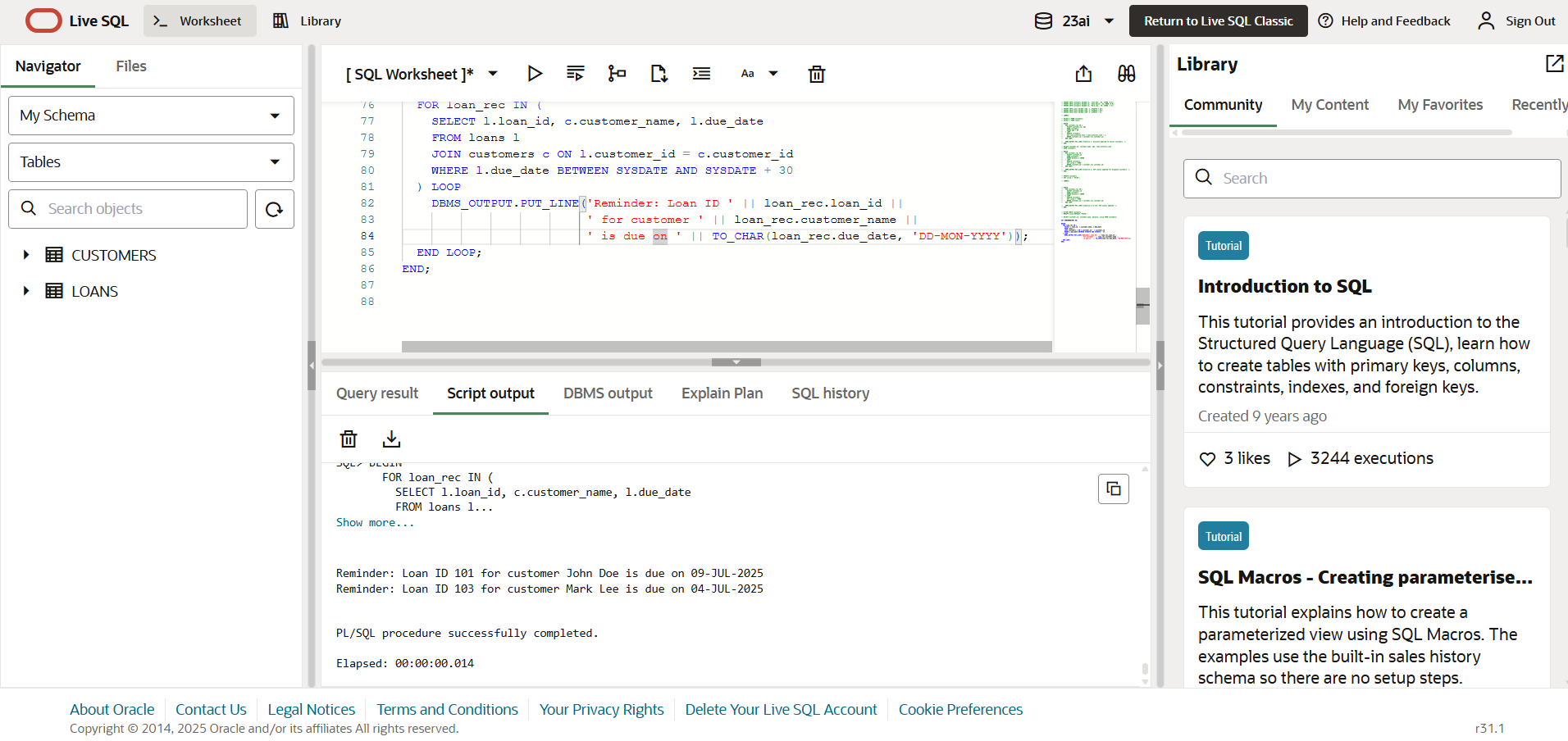
DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ID ' || loan\_rec.loan\_id ||

' for customer ' || loan\_rec.customer\_name ||

' is due on ' || TO\_CHAR(loan\_rec.due\_date, 'DD-MON-YYYY'));

END LOOP;

END;



**PL\_SQL**

EXERCISE 3 : STORED PROCEDURES

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE savings\_accounts';

EXECUTE IMMEDIATE 'DROP TABLE employees';

EXECUTE IMMEDIATE 'DROP TABLE bank\_accounts';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

CREATE TABLE savings\_accounts (

account\_id NUMBER PRIMARY KEY,

customer\_id NUMBER,

balance NUMBER

);

CREATE TABLE employees (

emp\_id NUMBER PRIMARY KEY,

name VARCHAR2(100),

department VARCHAR2(50),

salary NUMBER

);

CREATE TABLE bank\_accounts (

account\_id NUMBER PRIMARY KEY,

customer\_id NUMBER,

balance NUMBER

);

INSERT INTO savings\_accounts VALUES (201, 1, 5000);

INSERT INTO savings\_accounts VALUES (202, 2, 10000);

INSERT INTO employees VALUES (1, 'Alice', 'Sales', 50000);

INSERT INTO employees VALUES (2, 'Bob', 'HR', 40000);

INSERT INTO employees VALUES (3, 'Charlie', 'Sales', 55000);

INSERT INTO bank\_accounts VALUES (101, 1, 1000); -- from account

INSERT INTO bank\_accounts VALUES (102, 2, 500); -- to account

COMMIT;

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

FOR acc IN (SELECT account\_id, balance FROM savings\_accounts) LOOP

UPDATE savings\_accounts

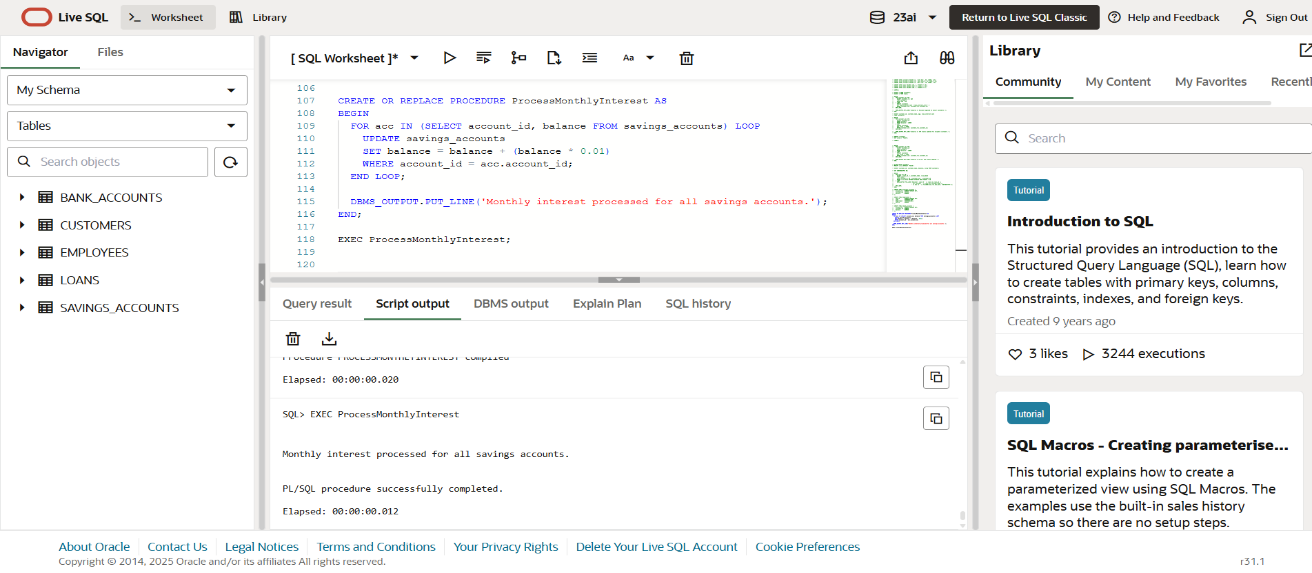
SET balance = balance + (balance \* 0.01)

WHERE account\_id = acc.account\_id;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed.');

END;



CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

dept\_name IN VARCHAR2,

bonus\_pct IN NUMBER

) AS

BEGIN

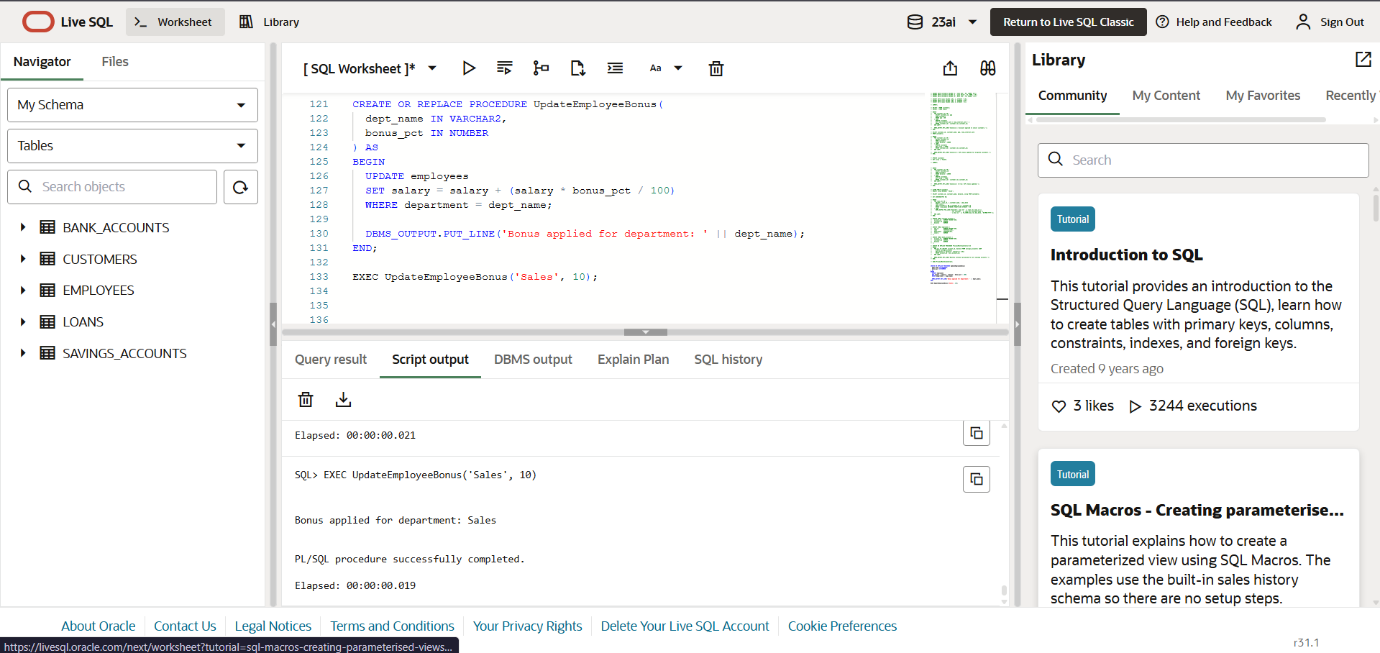
UPDATE employees

SET salary = salary + (salary \* bonus\_pct / 100)

WHERE department = dept\_name;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to department: ' || dept\_name);

END;



CREATE OR REPLACE PROCEDURE TransferFunds(

from\_account\_id IN NUMBER,

to\_account\_id IN NUMBER,

amount IN NUMBER

) AS

from\_balance NUMBER;

BEGIN

SELECT balance INTO from\_balance

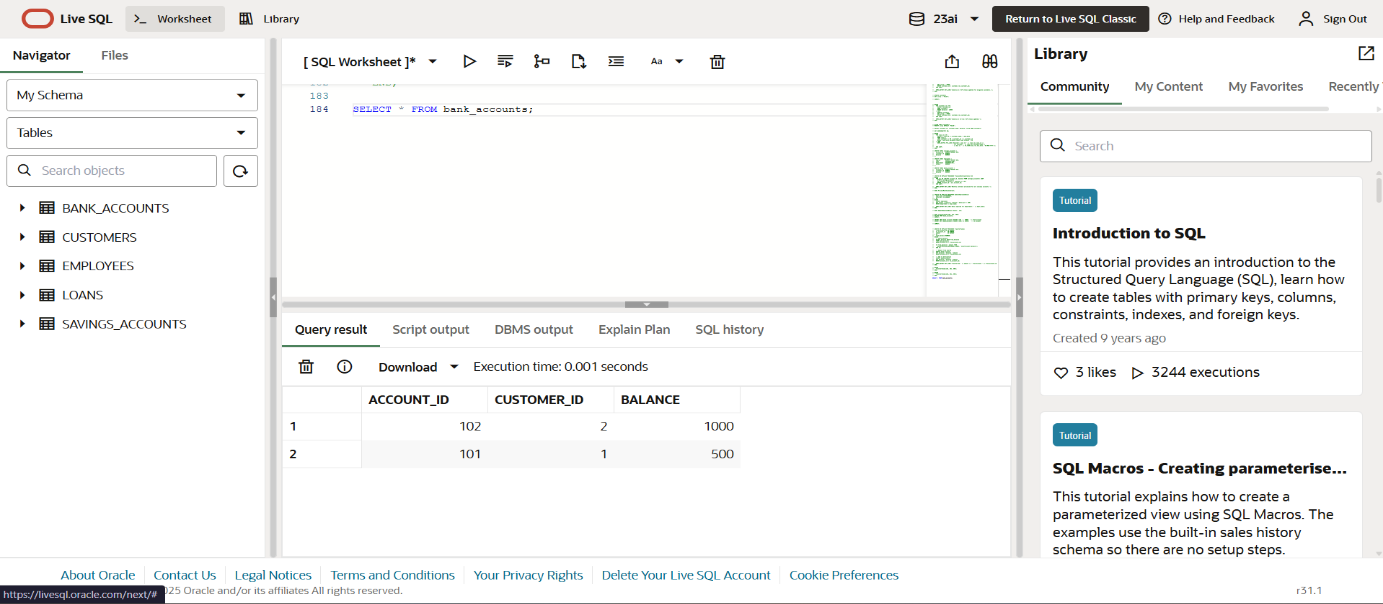
FROM bank\_accounts

WHERE account\_id = from\_account\_id;

IF from\_balance < amount THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance');

END IF;



UPDATE bank\_accounts

SET balance = balance - amount

WHERE account\_id = from\_account\_id;

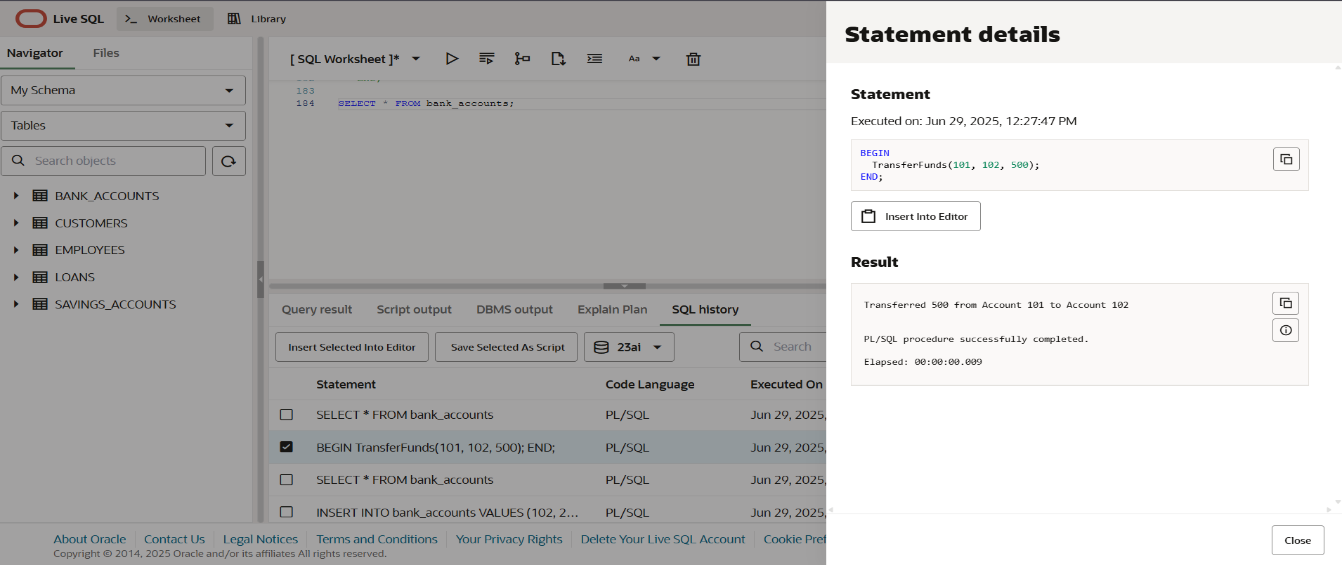
UPDATE bank\_accounts

SET balance = balance + amount

WHERE account\_id = to\_account\_id;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || amount || ' from Account ' || from\_account\_id || ' to Account ' || to\_account\_id);

END;



SET SERVEROUTPUT ON;

BEGIN

ProcessMonthlyInterest;

END;

BEGIN

UpdateEmployeeBonus('Sales', 10);

END;

BEGIN

TransferFunds(101, 102, 500);

END;

Final Output Checks :

SELECT \* FROM savings\_accounts;

SELECT \* FROM employees;

SELECT \* FROM bank\_accounts;