

PL/SQL EXERCISES

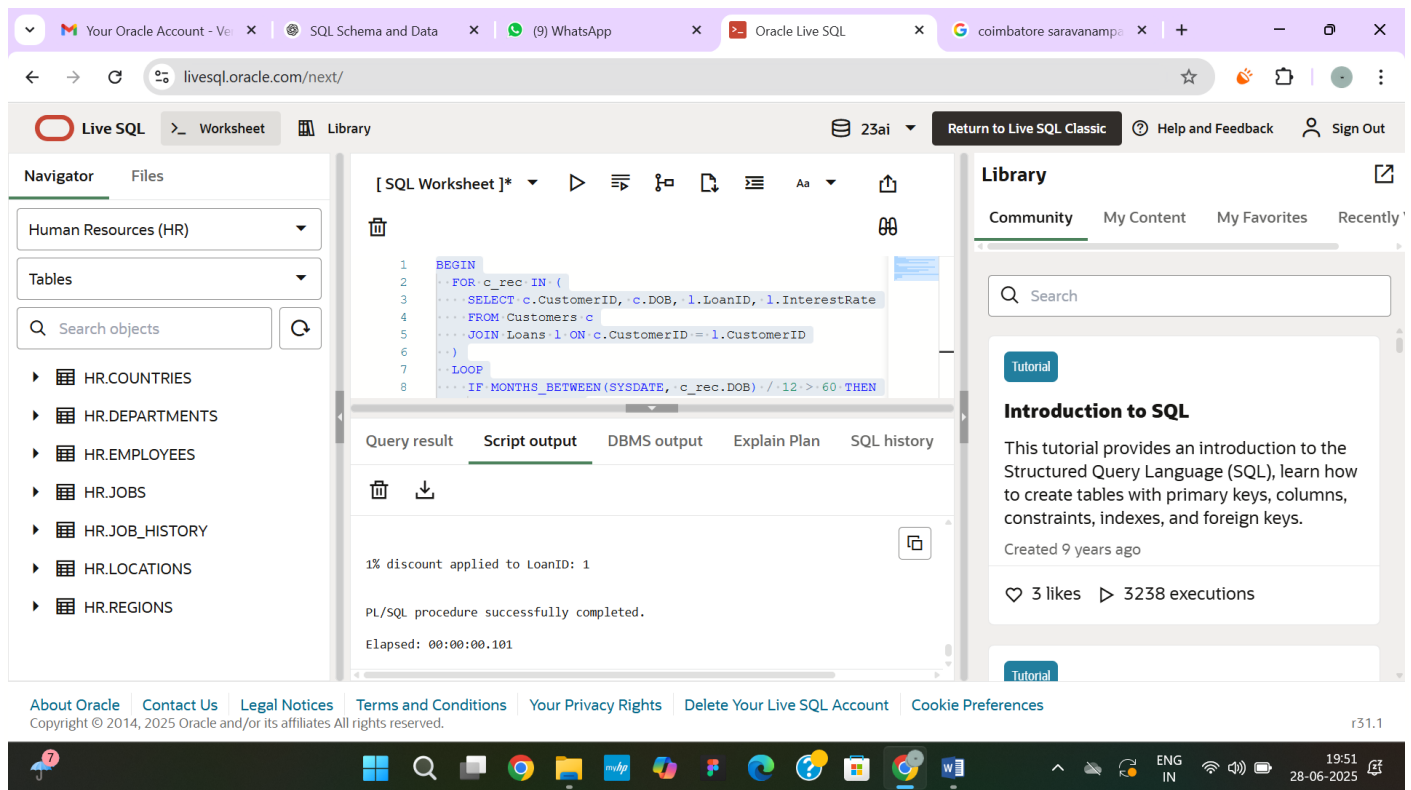
Exercise 1: Control Structures

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

```
BEGIN
FOR c_rec IN (
  SELECT c.CustomerID, c.DOB, l.LoanID, l.InterestRate
  FROM Customers c
  JOIN Loans l ON c.CustomerID = l.CustomerID
)
LOOP
  IF MONTHS_BETWEEN(SYSDATE, c_rec.DOB) / 12 > 60 THEN
    UPDATE Loans
    SET InterestRate = c_rec.InterestRate - 1
    WHERE LoanID = c_rec.LoanID;

    DBMS_OUTPUT.PUT_LINE('1% discount applied to LoanID: ' || c_rec.LoanID);
  END IF;
END LOOP;
END;
```

Output:



The screenshot displays the Oracle Live SQL web interface. The top navigation bar includes links for 'Your Oracle Account', 'SQL Schema and Data', 'WhatsApp', 'Oracle Live SQL', and a search bar. The main workspace is divided into three panels. The left panel, 'Navigator', shows a tree view of the 'Human Resources (HR)' schema, including tables like 'HR.COUNTRIES', 'HR.DEPARTMENTS', 'HR.EMPLOYEES', 'HR.JOBS', 'HR.JOB_HISTORY', 'HR.LOCATIONS', and 'HR.REGIONS'. The central panel, '[SQL Worksheet]*', contains the PL/SQL script for Scenario 1. Below the script, the 'Script output' tab is active, showing the execution results: '1% discount applied to LoanID: 1', 'PL/SQL procedure successfully completed.', and 'Elapsed: 00:00:00.101'. The right panel, 'Library', features a search bar and a list of tutorials, with 'Introduction to SQL' highlighted. The bottom of the interface shows a footer with legal notices and a system tray with the date '28-06-2025' and time '19:51'.

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

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

    DBMS_OUTPUT.PUT_LINE('Customer ID ' || c_rec.CustomerID || ' promoted to VIP.');
```

```

ELSE
    UPDATE Customers
    SET IsVIP = 'FALSE'
    WHERE CustomerID = c_rec.CustomerID;
END IF;
END LOOP;
END;
SELECT CustomerID, Name, Balance, IsVIP FROM Customers;

```

OUTPUT:

The screenshot shows the Oracle Live SQL web interface. The main window displays the query result for the query: `SELECT CustomerID, Name, Balance, IsVIP FROM Customers;`. The result is a table with 2 rows and 4 columns: CUSTOMERID, NAME, BALANCE, and ISVIP.

	CUSTOMERID	NAME	BALANCE	ISVIP
1	1	John Doe	1000	FALSE
2	2	Jane Smith	1500	FALSE

The interface also shows a sidebar with a Navigator pane listing database objects (HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB_HISTORY, HR.LOCATIONS, HR.REGIONS) and a Library pane with a search bar and a list of tutorials, including "Introduction to SQL".

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

```

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;

SELECT LoanID, CustomerID, EndDate
FROM Loans
WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30;

```

OUTPUT:

The screenshot displays the Oracle Live SQL web interface. The browser tabs include 'Your Oracle Account', 'SQL Schema and Data', '(9) WhatsApp', 'Oracle Live SQL', and 'coimbatore saravanamp'. The address bar shows 'livesql.oracle.com/next/'. The interface has a top navigation bar with 'Live SQL', 'Worksheet', and 'Library' tabs. The left sidebar contains a 'Navigator' with 'Files' and 'Tables' sections, listing objects like 'HR.COUNTRIES', 'HR.DEPARTMENTS', 'HR.EMPLOYEES', 'HR.JOBS', 'HR.JOB_HISTORY', 'HR.LOCATIONS', and 'HR.REGIONS'. The main area shows a SQL query in a worksheet:

```
1 SELECT LoanID, CustomerID, EndDate
2 FROM Loans
3 WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30;
4
```

Below the query, the 'Query result' tab is active, showing a table with columns 'LOANID', 'CUSTOMERID', and 'ENDDATE'. The table is empty, with the message 'No items to display.' and an execution time of 0.005 seconds. The right sidebar features a 'Library' section with a search bar and a tutorial titled 'Introduction to SQL', which describes the tutorial's purpose and shows 3 likes and 3238 executions. The bottom of the interface includes a footer with links like 'About Oracle', 'Contact Us', and 'Legal Notices', and a system tray at the very bottom showing the taskbar and system clock (20:05, 28-06-2025).

Exercise 2: Error Handling

Scenario 1: Handle exceptions during fund transfers between accounts.

```
CREATE OR REPLACE PROCEDURE SafeTransferFunds(
    p_FromAccountID IN NUMBER,
    p_ToAccountID IN NUMBER,
    p_Amount IN NUMBER
) IS
    v_FromBalance NUMBER;
    v_ErrorMsg VARCHAR2(4000);
BEGIN
    -- Lock the source account and get the balance
    SELECT Balance INTO v_FromBalance
    FROM Accounts
    WHERE AccountID = p_FromAccountID
    FOR UPDATE;

    -- Check for sufficient funds
    IF v_FromBalance < p_Amount THEN
        RAISE_APPLICATION_ERROR(-20001, 'Insufficient funds in the source account.');
```

END IF;

```
    -- Deduct from source account
    UPDATE Accounts
    SET Balance = Balance - p_Amount
    WHERE AccountID = p_FromAccountID;

    -- Add to destination account
    UPDATE Accounts
    SET Balance = Balance + p_Amount
    WHERE AccountID = p_ToAccountID;

    COMMIT;
```

```

DBMS_OUTPUT.PUT_LINE('Transfer successful.');
```

```

EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;

    v_ErrorMsg := SQLERRM;

    INSERT INTO TransferErrors (ErrorMessage)
    VALUES (v_ErrorMsg);

    DBMS_OUTPUT.PUT_LINE('Transfer failed. Error logged: ' || v_ErrorMsg);
END;
BEGIN
  SafeTransferFunds(1, 2, 500);
END;

BEGIN
  SafeTransferFunds(1, 2, 999999);
END;
SELECT * FROM TransferErrors ORDER BY ErrorTime DESC;
```

OUTPUT:

The screenshot shows the Oracle Live SQL interface. The main window displays the query result for the query: `SELECT * FROM TransferErrors ORDER BY ErrorTime DESC;`. The result shows one error entry:

ERRORID	ERRORMESSAGE	ERRORTIME
1	ORA-20001: Insuffici	2025-06-28T14:44:5

The interface also shows a sidebar with a Navigator and a Library section. The Library section features a tutorial titled "Introduction to SQL" with 3 likes and 3238 executions.

Scenario 2: Manage errors when updating employee salaries.

```

CREATE OR REPLACE PROCEDURE UpdateSalary(
  p_EmployeeID IN NUMBER,
  p_Percent IN NUMBER
) IS
  v_Salary Employees.Salary%TYPE;
  v_ErrorMessage VARCHAR2(4000);
BEGIN
  -- Try to fetch employee salary to verify existence
  SELECT Salary INTO v_Salary
  FROM Employees
  WHERE EmployeeID = p_EmployeeID
```

```

FOR UPDATE;

-- Update salary by given percentage
UPDATE Employees
SET Salary = Salary + (Salary * p_Percent / 100)
WHERE EmployeeID = p_EmployeeID;

COMMIT;

DBMS_OUTPUT.PUT_LINE('Salary updated successfully for Employee ID: ' || p_EmployeeID);

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        v_ErrorMessage := 'Employee ID ' || p_EmployeeID || ' does not exist.';
        INSERT INTO TransferErrors (ErrorMessage) VALUES (v_ErrorMessage);
        DBMS_OUTPUT.PUT_LINE('Error: ' || v_ErrorMessage);

    WHEN OTHERS THEN
        ROLLBACK;
        v_ErrorMessage := 'Unexpected error for Employee ID ' || p_EmployeeID || ': ' ||
SQLERRM;
        INSERT INTO TransferErrors (ErrorMessage) VALUES (v_ErrorMessage);
        DBMS_OUTPUT.PUT_LINE('Error: ' || v_ErrorMessage);

END;
BEGIN
    UpdateSalary(1, 10);
END;
BEGIN
    UpdateSalary(999, 10);
END;
SELECT * FROM Employees WHERE EmployeeID = 1;
SELECT * FROM TransferErrors ORDER BY ErrorTime DESC;

```

OUTPUT:

The screenshot shows the Oracle Live SQL interface. The main window displays a SQL worksheet with the following query:

```
SELECT * FROM TransferErrors ORDER BY ErrorTime DESC;
```

The query results are displayed in a table with the following columns: ERRORID, ERRORMESSAGE, and ERRORTIME. The results show two rows of errors.

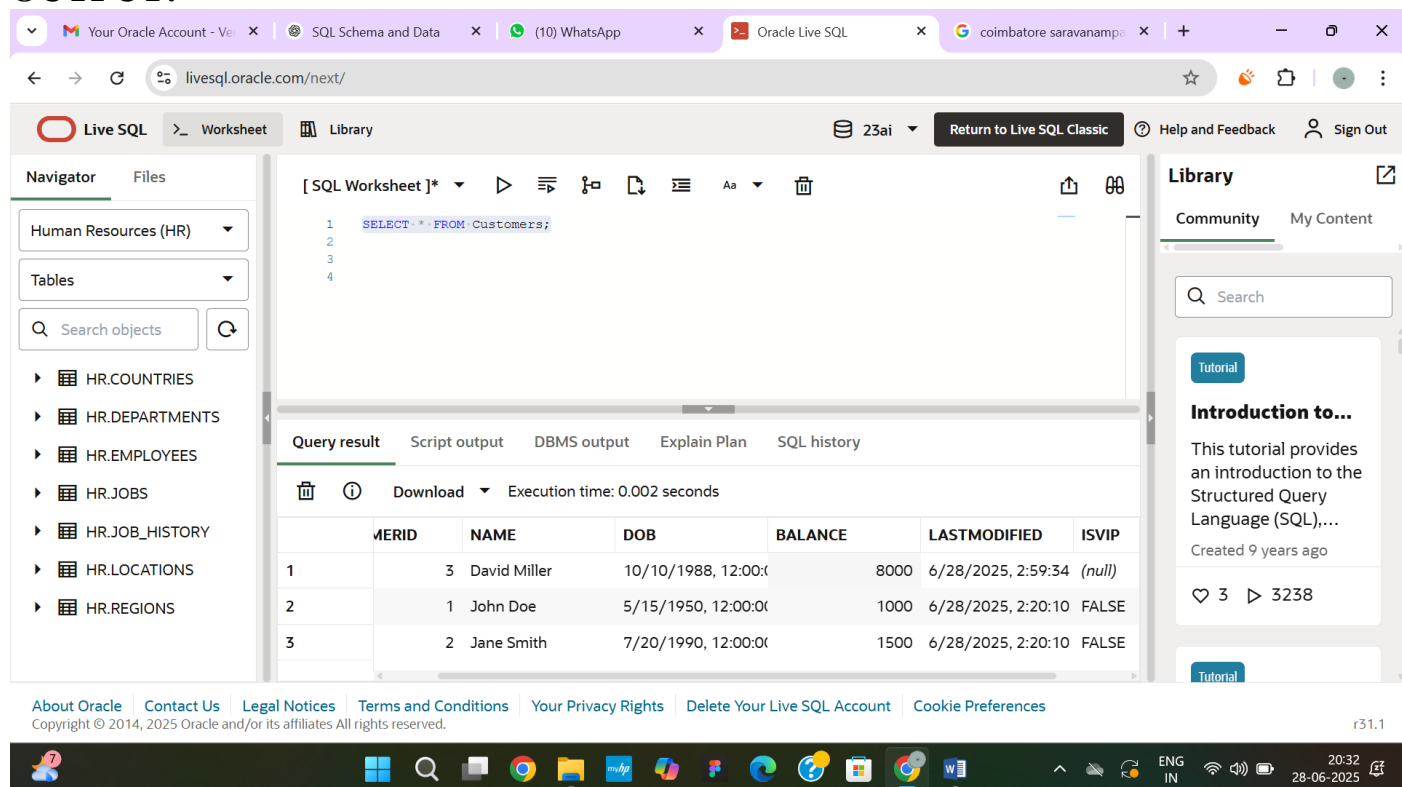
ERRORID	ERRORMESSAGE	ERRORTIME
1	Employee ID 999 doe	2025-06-28T14:53:00
2	ORA-20001: Insuffici	2025-06-28T14:44:50

The interface also includes a Navigator panel on the left showing the database schema (Human Resources (HR)) and a Library panel on the right showing a tutorial titled "Introduction to SQL".

Scenario 3: Ensure data integrity when adding a new customer.

```
CREATE OR REPLACE PROCEDURE AddNewCustomer (  
    p_CustomerID IN NUMBER,  
    p_Name IN VARCHAR2,  
    p_DOB IN DATE,  
    p_Balance IN NUMBER  
) IS  
    v_ErrorMessage VARCHAR2(4000);  
BEGIN  
    -- Attempt to insert the new customer  
    INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)  
    VALUES (p_CustomerID, p_Name, p_DOB, p_Balance, SYSDATE);  
  
    COMMIT;  
  
    DBMS_OUTPUT.PUT_LINE('Customer added successfully: ID ' || p_CustomerID);  
  
EXCEPTION  
    WHEN DUP_VAL_ON_INDEX THEN  
        v_ErrorMessage := 'Customer ID ' || p_CustomerID || ' already exists. Insertion  
prevented.';  
        INSERT INTO TransferErrors (ErrorMessage) VALUES (v_ErrorMessage);  
        DBMS_OUTPUT.PUT_LINE('Error: ' || v_ErrorMessage);  
  
    WHEN OTHERS THEN  
        ROLLBACK;  
        v_ErrorMessage := 'Unexpected error while adding customer ID ' || p_CustomerID || ': '  
|| SQLERRM;  
        INSERT INTO TransferErrors (ErrorMessage) VALUES (v_ErrorMessage);  
        DBMS_OUTPUT.PUT_LINE('Error: ' || v_ErrorMessage);  
END;  
BEGIN  
    AddNewCustomer(3, 'David Miller', TO_DATE('1988-10-10', 'YYYY-MM-DD'), 8000);  
END;  
BEGIN  
    AddNewCustomer(3, 'Duplicate Entry', TO_DATE('1990-01-01', 'YYYY-MM-DD'), 9000);  
END;
```

OUTPUT:



The screenshot displays the Oracle Live SQL interface. The central SQL Worksheet shows the query `SELECT * FROM Customers;` executed. Below the query, the Query result table is displayed, showing the following data:

CUSTID	NAME	DOB	BALANCE	LASTMODIFIED	ISVIP
1	David Miller	10/10/1988, 12:00:00	8000	6/28/2025, 2:59:34	(null)
2	John Doe	5/15/1950, 12:00:00	1000	6/28/2025, 2:20:10	FALSE
3	Jane Smith	7/20/1990, 12:00:00	1500	6/28/2025, 2:20:10	FALSE

The interface also includes a Navigator on the left with a search bar and a Library on the right with a search bar and a list of tutorials. The bottom of the interface shows the Oracle logo, contact information, and a copyright notice for 2014, 2025 Oracle and/or its affiliates.

Exercise 3: Stored Procedures

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

```
CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS
BEGIN
    FOR acc_rec IN (
        SELECT AccountID, Balance
        FROM Accounts
        WHERE AccountType = 'Savings'
        FOR UPDATE
    ) LOOP
        -- Apply 1% interest
        UPDATE Accounts
        SET Balance = acc_rec.Balance + (acc_rec.Balance * 0.01)
        WHERE AccountID = acc_rec.AccountID;

        DBMS_OUTPUT.PUT_LINE('1% interest applied to Account ID: ' || acc_rec.AccountID);
    END LOOP;

    COMMIT;
END;

SELECT * FROM Accounts WHERE AccountType = 'Savings';
```

OUTPUT:

The screenshot displays the Oracle Live SQL interface. The main window shows the SQL query: `SELECT * FROM Accounts WHERE AccountType = 'Savings';`. Below the query, the 'Query result' tab is active, showing a table with the following data:

ACCOUNTID	CUSTOMERID	ACCOUNTTYPE	BALANCE	LASTMODIFIED
1	1	Savings	505	6/28/2025, 2:20:10

The interface also includes a Navigator on the left with a search bar and a list of objects (HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB_HISTORY, HR.LOCATIONS, HR.REGIONS). The Library on the right shows a search bar and a list of tutorials, including 'Introduction to...'. The bottom status bar shows the time as 20:37 on 28-06-2025.

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

```
CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (
    p_Department IN VARCHAR2,
    p_BonusPercent IN NUMBER
) IS
BEGIN
    FOR emp_rec IN (
        SELECT EmployeeID, Salary
        FROM Employees
        WHERE Department = p_Department
        FOR UPDATE
    ) LOOP
```

```

-- Update salary with bonus
UPDATE Employees
SET Salary = emp_rec.Salary + (emp_rec.Salary * p_BonusPercent / 100)
WHERE EmployeeID = emp_rec.EmployeeID;

DBMS_OUTPUT.PUT_LINE('Bonus applied to Employee ID: ' || emp_rec.EmployeeID);
END LOOP;

COMMIT;
END;
BEGIN
  UpdateEmployeeBonus('IT', 10);
END;
SELECT * FROM Employees WHERE Department = 'IT';

```

OUTPUT:

The screenshot shows the Oracle Live SQL interface. The query executed is `SELECT * FROM Employees WHERE Department = 'IT';`. The result shows one employee: Bob Brown, Developer, with a salary of 66000, in the IT department. The interface includes a Navigator on the left with a tree view of database objects (HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB_HISTORY, HR.LOCATIONS, HR.REGIONS). The right sidebar shows a Library with a search bar and a tutorial titled "Introduction to..." which provides an introduction to the Structured Query Language (SQL). The bottom of the interface shows a footer with links to About Oracle, Contact Us, Legal Notices, Terms and Conditions, Your Privacy Rights, Delete Your Live SQL Account, and Cookie Preferences. The copyright notice is "Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved." and the version is "r31.1".

EMPLOYEEID	NAME	POSITION	SALARY	DEPARTMENT
2	Bob Brown	Developer	66000	IT

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

```

CREATE OR REPLACE PROCEDURE TransferFunds(
  p_FromAccountID IN NUMBER,
  p_ToAccountID IN NUMBER,
  p_Amount IN NUMBER
) IS
  v_FromBalance NUMBER;
BEGIN
  IF p_FromAccountID = p_ToAccountID THEN
    RAISE_APPLICATION_ERROR(-20001, 'Source and destination accounts must be different.');
```

END IF;

```

  SELECT Balance INTO v_FromBalance
  FROM Accounts
  WHERE AccountID = p_FromAccountID
  FOR UPDATE;
  IF v_FromBalance < p_Amount THEN
    RAISE_APPLICATION_ERROR(-20002, 'Insufficient funds in the source account.');
```

END IF;

```

  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('Successfully transferred ' || p_Amount || ' from Account ' ||
p_FromAccountID || ' to Account ' || p_ToAccountID);
EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('Transfer failed: ' || SQLERRM);
END;
BEGIN
    TransferFunds(1, 2, 300);
END;

```

OUTPUT:

The screenshot displays the Oracle Live SQL web interface. The browser tabs include 'Your Oracle Account - Ver...', '(7) WhatsApp', 'SQL Schema and Data', 'Oracle Live SQL', and 'coimbatore saravanamp...'. The address bar shows 'livesql.oracle.com/next/'.

The interface features a top navigation bar with 'Live SQL', 'Worksheet', and 'Library' tabs. Below this is a 'Navigator' panel on the left with a 'Files' section containing 'Human Resources (HR)' and 'Tables'. A search bar for objects is also present.

The main workspace shows a SQL script in a text editor:

```

1 BEGIN
2   TransferFunds(1, 2, 300);
3 END;
4
5

```

Below the editor, the 'Script output' tab is active, displaying the following results:

```

END;

Successfully transferred 300 from Account 1 to Account 2

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.010

```

On the right side, there is a 'Library' panel with a search bar and a list of items. The first item is a tutorial titled 'Introduction to...' with a description: 'This tutorial provides an introduction to the Structured Query Language (SQL),...'. It was created 9 years ago and has 3 likes and 3238 views.

At the bottom, there is a footer with links: 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', 'Delete Your Live SQL Account', and 'Cookie Preferences'. The copyright notice states: 'Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved.' The version number 'r31.1' is also visible.

