Parameterized Cursor PL/SQL Practical: Merge RollCall Tables

```
-- File: merge_rollcall_using_parameterized_cursor.sql
-- Purpose: Create two tables (O_RollCall and N_RollCall), insert sample data,
-- and use a PL/SQL block with a parameterized cursor to merge N_RollCall into O_RollCall
-- skipping any rows that already exist (based on Roll_no and Name).
-- Run in Oracle APEX SQL Workshop -> SQL Commands (Run Script)

-- 0) Drop tables if they exist (safe in dev/test)
BEGIN
  BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE N_RollCall CASCADE CONSTRAINTS';
  EXCEPTION WHEN OTHERS THEN
    IF SQLCODE != -942 THEN RAISE; END IF; -- ignore "table does not exist"
  END;
  BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE O_RollCall CASCADE CONSTRAINTS';
  EXCEPTION WHEN OTHERS THEN
    IF SQLCODE != -942 THEN RAISE; END IF;
  END;
END;
/
-- 1) Create target table O_RollCall and source table N_RollCall
CREATE TABLE O_RollCall (
  Roll_no NUMBER PRIMARY KEY,
  Name    VARCHAR2(100)
);
/

CREATE TABLE N_RollCall (
  Roll_no NUMBER,
  Name    VARCHAR2(100)
);
/

-- 2) Insert sample data into O_RollCall (existing data)
INSERT ALL
  INTO O_RollCall (Roll_no, Name) VALUES (1, 'Himanshu')
  INTO O_RollCall (Roll_no, Name) VALUES (2, 'Ram')
  INTO O_RollCall (Roll_no, Name) VALUES (3, 'Soham')
SELECT * FROM dual;
/
COMMIT;
/

-- 3) Insert sample data into N_RollCall (new incoming data; some duplicates, some new)
INSERT ALL
  INTO N_RollCall (Roll_no, Name) VALUES (2, 'Ram')          -- duplicate
  INTO N_RollCall (Roll_no, Name) VALUES (3, 'Soham')        -- duplicate
  INTO N_RollCall (Roll_no, Name) VALUES (4, 'Mohan')        -- new
  INTO N_RollCall (Roll_no, Name) VALUES (5, 'Om')           -- new
  INTO N_RollCall (Roll_no, Name) VALUES (6, 'Aman Kumar')   -- new
SELECT * FROM dual;
/
COMMIT;
/

-- 4) PL/SQL block: Use an explicit cursor to iterate N_RollCall and a parameterized cursor
--    to check for existence in O_RollCall. Insert only if not present.
DECLARE
  CURSOR cur_new IS
    SELECT Roll_no, Name FROM N_RollCall;

  CURSOR cur_exists(p_roll_no IN NUMBER, p_name IN VARCHAR2) IS
    SELECT Roll_no
      FROM O_RollCall
     WHERE Roll_no = p_roll_no
       AND LOWER(Name) = LOWER(p_name);

  v_roll  N_RollCall.Roll_no%TYPE;
  v_name  N_RollCall.Name%TYPE;
  v_count NUMBER := 0;
```

```
  v_inserted NUMBER := 0;
BEGIN
  v_count := 0;
  v_inserted := 0;

  OPEN cur_new;
  LOOP
    FETCH cur_new INTO v_roll, v_name;
    EXIT WHEN cur_new%NOTFOUND;

    v_count := v_count + 1;

    OPEN cur_exists(v_roll, v_name);

    DECLARE
      v_dummy O_RollCall.Roll_no%TYPE;
    BEGIN
      FETCH cur_exists INTO v_dummy;
      IF cur_exists%NOTFOUND THEN
        INSERT INTO O_RollCall (Roll_no, Name) VALUES (v_roll, v_name);
        v_inserted := v_inserted + 1;
        DBMS_OUTPUT.PUT_LINE('Inserted: Roll=' || v_roll || ', Name=' || v_name);
      ELSE
        DBMS_OUTPUT.PUT_LINE('Skipped (already exists): Roll=' || v_roll || ', Name=' || v_name);
      END IF;
    END;

    CLOSE cur_exists;
  END LOOP;
  CLOSE cur_new;

  COMMIT;

  DBMS_OUTPUT.PUT_LINE('Total rows processed: ' || v_count);
  DBMS_OUTPUT.PUT_LINE('Total rows inserted: ' || v_inserted);
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;
    DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
END;
/
-- 5) Verify final state of O_RollCall
SELECT * FROM O_RollCall ORDER BY Roll_no;
/
-- 6) Optionally: show N_RollCall too
SELECT * FROM N_RollCall ORDER BY Roll_no;
/
```