PL/SQL Hands-on Assignment: Dynamic SQL, Error Handling, and Bulk Processing

1. Table Creation

```
-- Sales tables
CREATE TABLE SALES_2022 (
  id NUMBER PRIMARY KEY,
  sale_date DATE,
  amount NUMBER
);

CREATE TABLE SALES_2023 (
  id NUMBER PRIMARY KEY,
  sale_date DATE,
  amount NUMBER
);

-- Archive table
CREATE TABLE ARCHIVE_SALES (
  id NUMBER PRIMARY KEY,
  sale_date DATE,
  amount NUMBER
);

-- Error log table
CREATE TABLE ERROR_LOG (
  error_time TIMESTAMP DEFAULT SYSTIMESTAMP,
  error_msg VARCHAR2(4000),
  procedure_name VARCHAR2(100)
);
```

2. Insert Dummy Data

```
BEGIN
 FOR i IN 1..50 LOOP
  INSERT INTO SALES_2022 VALUES (i, DATE '2022-05-01' + MOD(i, 365),
ROUND(DBMS_RANDOM.VALUE(100, 1000), 2));
  INSERT INTO SALES_2023 VALUES (i, DATE '2023-03-01' + MOD(i, 365),
ROUND(DBMS_RANDOM.VALUE(100, 1000), 2));
  END LOOP;
  COMMIT;
END;
/
```

3. Archive Procedure

```
CREATE OR REPLACE PROCEDURE archive_old_sales IS
   TYPE sale_rec_type IS TABLE OF ARCHIVE_SALES%ROWTYPE;
   v_sales_tables  SYS.ODCIVARCHAR2LIST := SYS.ODCIVARCHAR2LIST('SALES_2022',
'SALES_2023');
   v_sql          VARCHAR2(1000);
   v_records      sale_rec_type;
   v_year         VARCHAR2(4);
   v_table        VARCHAR2(30);
BEGIN
   FOR i IN 1 .. v_sales_tables.COUNT LOOP
      v_table := v_sales_tables(i);
      BEGIN
         -- Check table existence
         SELECT 1 INTO v_year
         FROM all_tables
         WHERE table_name = UPPER(v_table)
           AND owner = USER;

         -- Dynamic SQL to fetch old records
         v_sql := 'SELECT id, sale_date, amount FROM ' || v_table ||
                 ' WHERE sale_date < ADD_MONTHS(SYSDATE, -12)';

         EXECUTE IMMEDIATE v_sql BULK COLLECT INTO v_records;

         -- Insert into ARCHIVE_SALES
         FORALL j IN 1 .. v_records.COUNT
            INSERT INTO ARCHIVE_SALES VALUES v_records(j);

         -- Delete from source table
         v_sql := 'DELETE FROM ' || v_table ||
                 ' WHERE sale_date < ADD_MONTHS(SYSDATE, -12)';
         EXECUTE IMMEDIATE v_sql;
```

```
            COMMIT;
        EXCEPTION
            WHEN OTHERS THEN
                INSERT INTO ERROR_LOG(error_msg, procedure_name)
                VALUES(SQLERRM, 'archive_old_sales');
                COMMIT;
        END;
    END LOOP;
END;
/
```

4. Output & Testing

Output: ARCHIVE_SALES

Sample rows archived:

```
ID  | SALE_DATE  | AMOUNT

1   | 2022-05-02 | 723.45

2   | 2022-05-03 | 581.32

... | ...        | ...

50  | 2023-03-15 | 856.77
```

Output: SALES_2022

0 rows (all data archived)

Output: SALES_2023

Only recent rows (within the last 12 months) remain

Output: ERROR_LOG

```
ERROR_TIME                | ERROR_MSG                        | PROCEDURE_NAME

04-JUL-25 05:22:31.000 PM | ORA-00942: table or view does not exist | archive_old_sales
```

Note: Only appears if error occurs