The bank manager has decided to activate all those accounts which were previously marked as inactive for performing no transaction in last 365 days. Write a PL/SQ block (using implicit cursor) to update the status of account, display an approximate message based on the no. of rows affected by the update. (Use of %FOUND, %NOTFOUND, %ROWCOUNT)
ANS=
PL/SQL Block

```
sql

Copy code

DECLARE

  -- Variable to hold the number of rows affected

  v_rows_updated INTEGER;

BEGIN

  -- Update statement to activate accounts marked as inactive

  UPDATE Accounts

  SET Status = 'Active'

  WHERE Status = 'Inactive'

   AND Last_Transaction_Date < SYSDATE - 365;


  -- Use SQL%ROWCOUNT to get the number of rows affected by the last DML statement

  v_rows_updated := SQL%ROWCOUNT;


  -- Display message based on the number of rows updated

  IF v_rows_updated > 0 THEN

    DBMS_OUTPUT.PUT_LINE(v_rows_updated || ' account(s) have been activated.');

  ELSE

    DBMS_OUTPUT.PUT_LINE('No accounts were found to activate.');

  END IF;


EXCEPTION

  WHEN OTHERS THEN

    -- Handle any unexpected errors

    DBMS_OUTPUT.PUT_LINE('Error occurred: ' || SQLERRM);

END;
```

/

Explanation of the PL/SQL Block

1. **Update Statement:**
    - The UPDATE statement activates all accounts that are marked as 'Inactive' and have not had any transactions in the last 365 days. This is checked using the condition Last_Transaction_Date < SYSDATE - 365.
2. **SQL%ROWCOUNT:**
    - After the UPDATE statement, SQL%ROWCOUNT is used to get the number of rows that were updated. This value is stored in the variable v_rows_updated.
3. **Conditional Message:**
    - An IF statement checks the value of v_rows_updated.
        - If it's greater than 0, it prints a message indicating how many accounts were activated.
        - If it's 0, it prints a message indicating that no accounts were found to activate.
4. **Exception Handling:**
    - An exception block catches any unexpected errors that may occur during the execution and outputs an error message using DBMS_OUTPUT.PUT_LINE.

Important Notes

- Ensure that you have a table named Accounts with the columns Status and Last_Transaction_Date. Adjust the table name and column names if they differ in your database.
- To see the output messages from DBMS_OUTPUT.PUT_LINE, make sure to enable the output in your SQL environment (e.g., SQL*Plus or SQL Developer).
- The code assumes that the status of accounts is either 'Active' or 'Inactive'. Adjust the status values as necessary for your specific implementation.