// Create new stage

CREATE OR REPLACE STAGE MANAGE\_DB.external\_stages.aws\_stage\_errorex

url='s3://bucketsnowflakes4';

// List files in stage

LIST @MANAGE\_DB.external\_stages.aws\_stage\_errorex;

// Create example table

CREATE OR REPLACE TABLE OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX (

ORDER\_ID VARCHAR(30),

AMOUNT INT,

PROFIT INT,

QUANTITY INT,

CATEGORY VARCHAR(30),

SUBCATEGORY VARCHAR(30));

// Demonstrating error message

COPY INTO OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX

FROM @MANAGE\_DB.external\_stages.aws\_stage\_errorex

file\_format= (type = csv field\_delimiter=',' skip\_header=1)

files = ('OrderDetails\_error.csv');

// Validating table is empty

SELECT \* FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX ;

// Error handling using the ON\_ERROR option,continue will skup the rows with error and will load the other rows because here the eroor limit is equal to no, rows.Eroor limit is nothing but signifies the no of rows can be loaded if error is <= error limit.

COPY INTO OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX

FROM @MANAGE\_DB.external\_stages.aws\_stage\_errorex

file\_format= (type = csv field\_delimiter=',' skip\_header=1)

files = ('OrderDetails\_error.csv')

ON\_ERROR = 'CONTINUE';

// Validating results and truncating table

SELECT \* FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

SELECT COUNT(\*) FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

//Truncate table before executing ON-ERROR on same table as rows are already inserted/loaded.

TRUNCATE TABLE OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

// Error handling using the ON\_ERROR option = ABORT\_STATEMENT (default), as soon as it encounters error it stops loading the rows.If one file has the erorr the other file will also be impacted and rows will get loaded.

COPY INTO OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX

FROM @MANAGE\_DB.external\_stages.aws\_stage\_errorex

file\_format= (type = csv field\_delimiter=',' skip\_header=1)

files = ('OrderDetails\_error.csv','OrderDetails\_error2.csv')

ON\_ERROR = 'ABORT\_STATEMENT';

// Validating results and truncating table

SELECT \* FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

SELECT COUNT(\*) FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

TRUNCATE TABLE OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

// Error handling using the ON\_ERROR option = SKIP\_FILE,same as above but it allows the row loading for the non-error file as well.

COPY INTO OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX

FROM @MANAGE\_DB.external\_stages.aws\_stage\_errorex

file\_format= (type = csv field\_delimiter=',' skip\_header=1)

files = ('OrderDetails\_error.csv','OrderDetails\_error2.csv')

ON\_ERROR = 'SKIP\_FILE';

// Validating results and truncating table

SELECT \* FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

SELECT COUNT(\*) FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

TRUNCATE TABLE OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

// Error handling using the ON\_ERROR option = SKIP\_FILE\_<number>, it works as continue.All rows will get executed if error limit is less than <number>(SKIP\_FILE\_<number>)

COPY INTO OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX

FROM @MANAGE\_DB.external\_stages.aws\_stage\_errorex

file\_format= (type = csv field\_delimiter=',' skip\_header=1)

files = ('OrderDetails\_error.csv','OrderDetails\_error2.csv')

ON\_ERROR = 'SKIP\_FILE\_2';

// Validating results and truncating table

SELECT \* FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

SELECT COUNT(\*) FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

TRUNCATE TABLE OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;

// Error handling using the ON\_ERROR option = SKIP\_FILE\_<number>

COPY INTO OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX

FROM @MANAGE\_DB.external\_stages.aws\_stage\_errorex

file\_format= (type = csv field\_delimiter=',' skip\_header=1)

files = ('OrderDetails\_error.csv','OrderDetails\_error2.csv')

ON\_ERROR = 'SKIP\_FILE\_3%';

SELECT \* FROM OUR\_FIRST\_DB.PUBLIC.ORDERS\_EX;