



## [Snowflake] 4-2. Bulk Loading

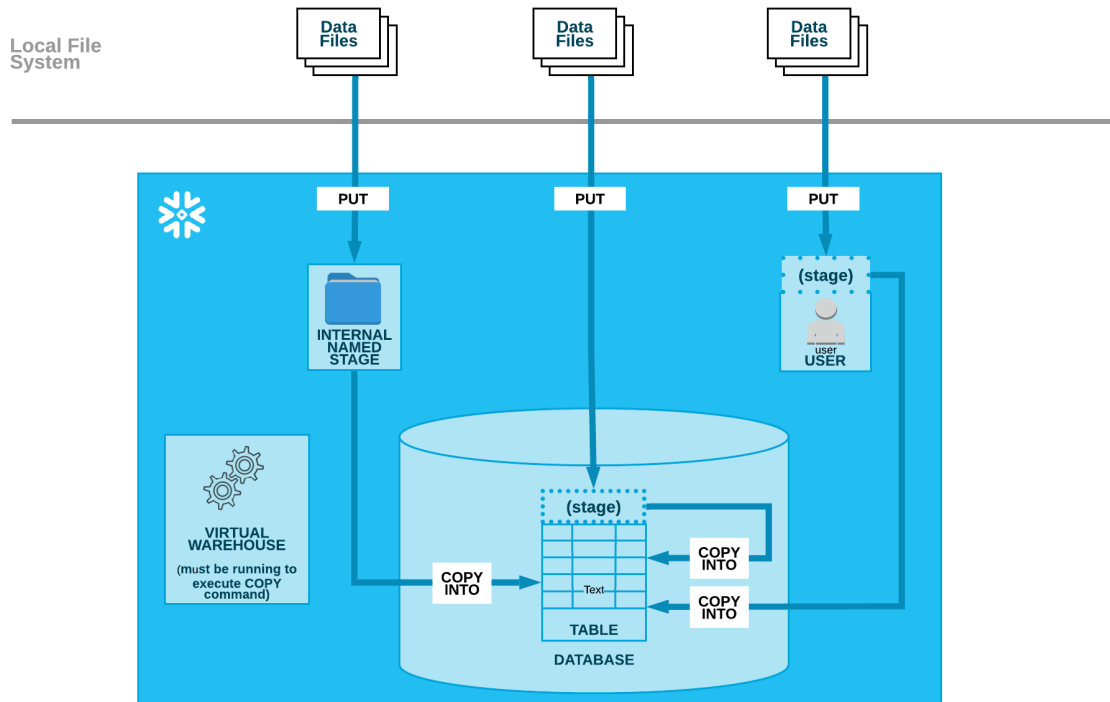


[노션 웹 공유 링크 \(댓글 & 상세설명 참고\)](#)

### References

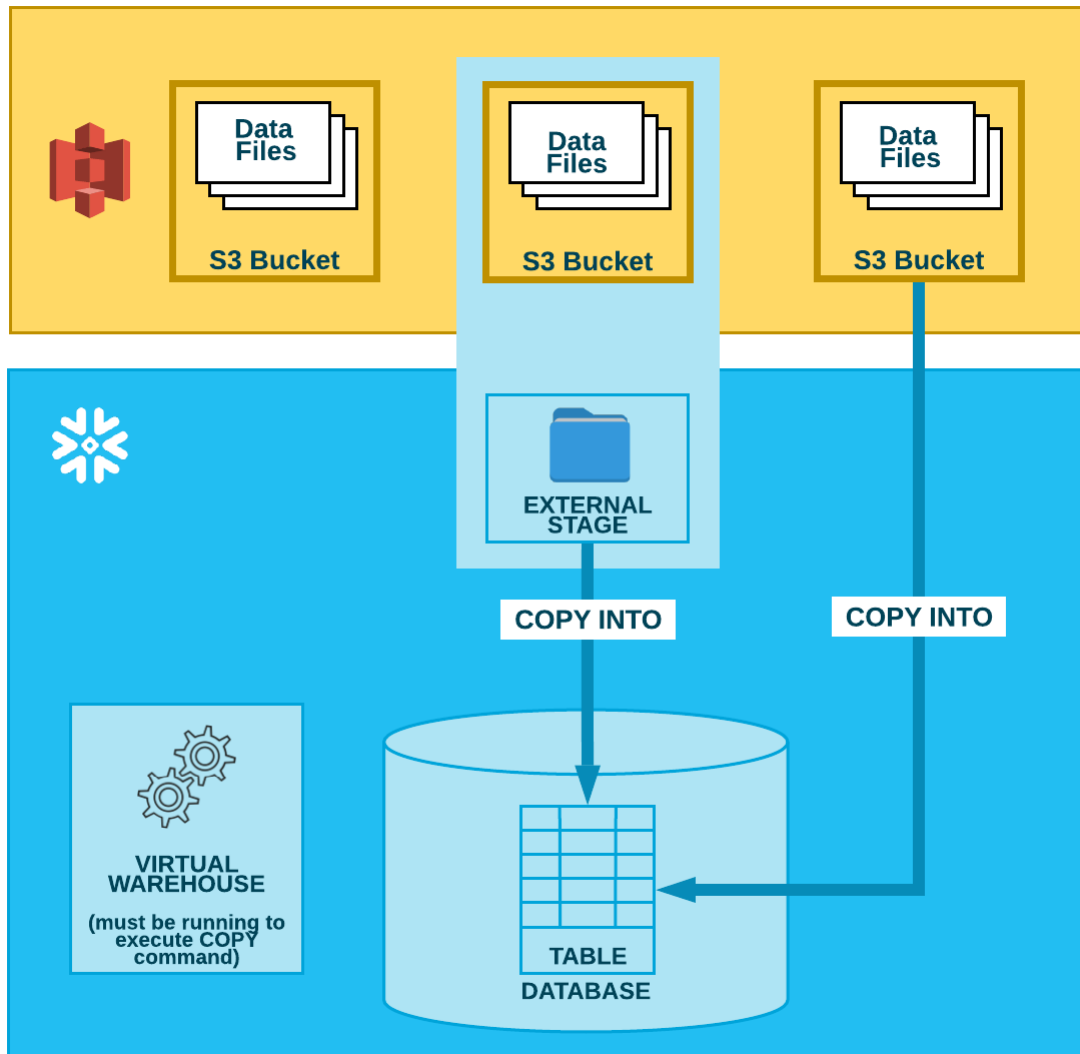
- [Snowflake Learn \(SnowPro PREP-CORE Course\)\\_4장 2강](#)
- [Snowflake 설명서 \(대량 로딩\)](#)

- 
- Bulk Loading (대량 로딩)

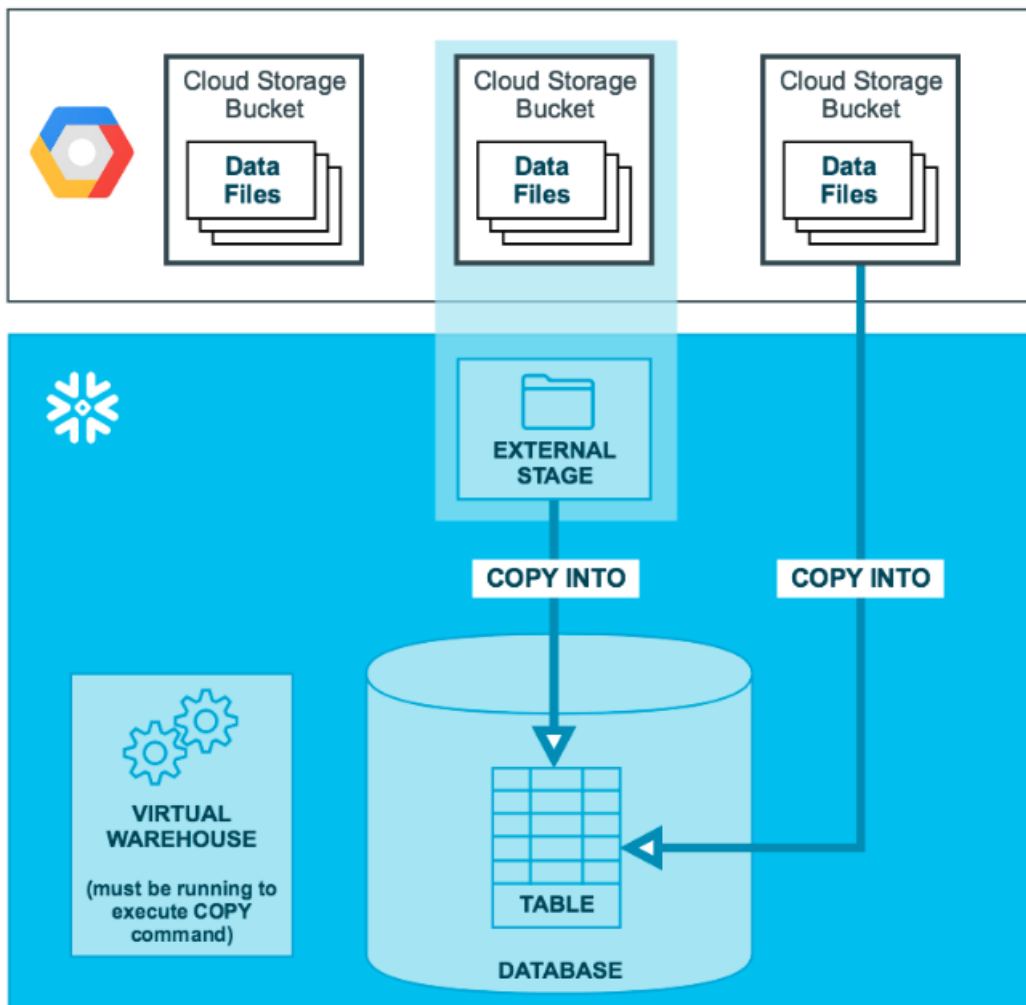


로컬 파일 시스템에서 대량 로드

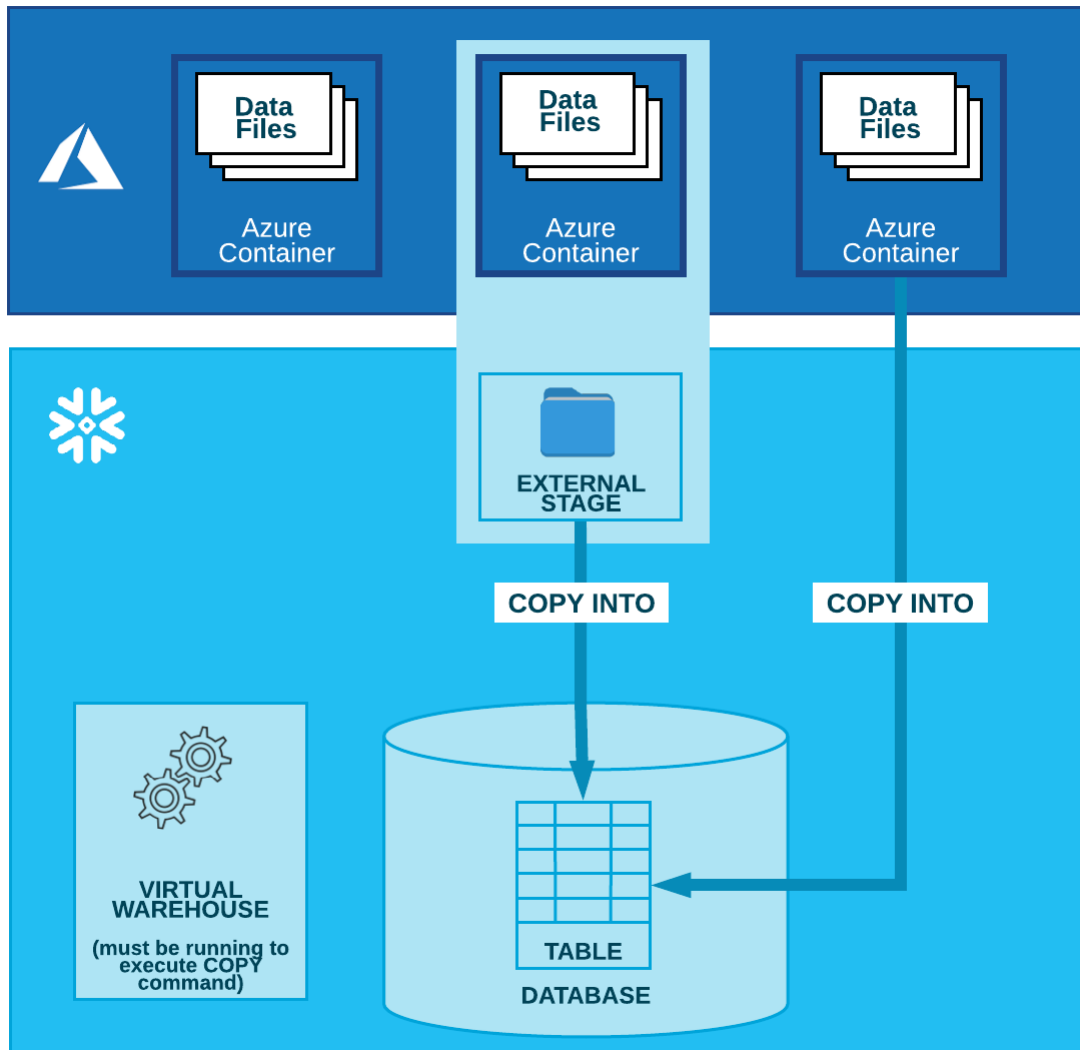
- Local File System 에서 대량 로드
  - PUT 명령을 사용하여 1개 이상의 데이터 파일을 Snowflake 스테이지에 업로드
  - COPY INTO <테이블> 명령을 사용하여 스테이징 된 파일의 내용을 Snowflake 데이터 베이스 테이블에 로드



AWS S3



Google Cloud Storage



Microsoft Azure

- Cloud Storage 에서 대량 로드
  - 데이터 파일이 Cloud Storage 내에 존재하기 때문에 PUT 명령이 필요 없음
  - 외부 스테이지를 생성 후 COPY INTO 명령으로 테이블에 데이터를 불러옴
    - 스테이지를 생성하고 로그인 방법, 데이터 암호 해독 방법 등을 지정, 권장 사항
  - Cloud Storage 에서 바로 COPY INTO 명령으로 테이블에 데이터를 불러옴
    - 단, 이 경우 COPY INTO 명령을 실행할 때 마다 위치, 자격증명 및 키를 지정해야 하기에 번거로움, 안전한 방법이 아님
  
- Local Storage 에서 대량 로딩 예시

## EXAMPLE FROM LOCAL STORAGE

Create internal stage

```
CREATE STAGE my_stage  
FILE_FORMAT = my_csv_format;
```

PUT data to stage

```
PUT FILE:///data/data.csv @my_stage;
```

COPY data to table

```
COPY INTO my_table FROM @my_stage;
```

Note: Compression during the PUT uses local resources. The local host needs sufficient memory, and space in /tmp, or the PUT will fail.

- Internal Stage 생성 → Stage에 데이터 PUT → 테이블에 데이터 COPY INTO
  - CREATE STAGE <STAGE 명> FILE\_FORMAT = <생성한 FILE\_FORMAT 명>
  - PUT FILE:///data/data.csv <STAGE 명>
  - COPY INTO <테이블 명> FROM <STAGE 명>
- Cloud Storage 에서 대량 로딩 예시

## EXAMPLE FROM CLOUD STORAGE

Create external stage  
(pointer to cloud storage location)

```
CREATE STAGE my_s3_stage  
url='s3://mybucket/encrypted_files/'  
CREDENTIALS=(aws_key_id='*****'  
aws_secret_key='*****')  
ENCRYPTION=(master_key = '*****')  
FILE_FORMAT = my_csv_format;
```

Load data using COPY

```
COPY INTO my_table  
FROM @my_s3_stage  
PATTERN='.*sales.*.csv';
```

- External Stage 생성 → 테이블에 데이터 COPY INTO
  - CREATE STAGE <STAGE 명> url=<external storage url> CREDENTIALS=<external storage 자격증명> ENCRYPTION=<마스터 키> FILE\_FORMAT = <FILE\_FORMAT 명>
  - COPY INTO <테이블명> FROM <STAGE 명>

- COPY INTO 명령 파라미터

## COPY INTO COMMAND PARAMETERS



Optional items in blue

```
COPY INTO [<namespace>].<table name>
FROM { <stage> | <external location> }

[FILES = ( '<file name>' [ , '<file name>' ] [ , ... ] )]

[PATTERN = '<regex pattern>']

[FILE_FORMAT = ( { FORMAT_NAME = '<namespace>.<format name>' |
                    TYPE = { CSV | JSON | AVRO | ORC | PARQUET | XML }
                    [ <format type options> } ] )]

[VALIDATION_MODE = RETURN_<n>_ROWS | RETURN_ERRORS | RETURN_ALL_ERRORS]

[<copy options>]
```

- VALIDATION\_MODE : 로드하지 않고 데이터를 확인하는데 사용, 데이터는 테이블에 로드되지 않음

## COPY INTO COMMAND OPTIONS



The COPY command has several options for handling data and errors (defaults are show in blue)

```
SIZE_LIMIT = <num bytes>

PURGE = TRUE | FALSE

RETURN_FAILED_ONLY = TRUE | FALSE

ENFORCE_LENGTH = TRUE | FALSE

TRUNCATECOLUMNS = TRUE | FALSE

FORCE = TRUE | FALSE

LOAD_UNCERTAIN_FILES = TRUE | FALSE

ON_ERROR = <error handling option>
```

- SIZE\_LIMIT : 데이터 양이 초과 시 데이터 로드 중지
- PURGE : 성공적으로 로드 된 모든 파일을 스테이지에서 삭제할지 여부 결정
- RETURN\_FAILED\_ONLY : 결과 집합에서 실패한 행만 반환
- ENFORCED\_LENGTH + TRUNCATEDCOLUMNS : 서로 반대 되는 옵션
  - TRUE / FALSE 로 설정 시 테이블의 데이터 길이와 맞지 않을 경우 에러 반환

- FALSE / TRUE 설정 시 테이블의 데이터 길이에 맞게 데이터가 잘리고, 처리가 성공적으로 완료 된 것으로 처리
- FORCE : 이미 로드한 파일을 강제로 다시 로드 가능
- LOAD\_UNCERTAIN\_FILES : 스테이지에 오래된 파일 (64일 후 메타데이터가 지워진 파일)을 로드할 때 사용하는 옵션

## ERROR HANDLING OPTIONS

➤

Use the ON\_ERROR option to control how errors in the data load are handled

Supported Values	Notes
CONTINUE	Continue loading the file.
SKIP_FILE	Skip file if any errors are encountered in the file.
SKIP_FILE_<num> (e.g. SKIP_FILE_10)	Skip file when the number of errors in the file is equal to or exceeds the specified number.
SKIP_FILE_<num>% (e.g. SKIP_FILE_10%)	Skip file when the percentage of errors in the file exceeds the specified percentage.
ABORT_STATEMENT (default)	Abort the COPY statement if any error is encountered.

- ON\_ERROR : 오류 처리 방법 지정
  - CONTINUE : 모든 단일 기록을 가능한 로드하려고 시도
  - SKIP\_FILE : 파일에서 오류가 발생 시 파일을 건너 뛰고 다음 파일로 이동, 숫자나 퍼센트로 지정하여 특정 수, 비율에 도달하면 건너뛰도록 지정 가능
  - ABORT\_STATEMENT (default) : 오류가 발생 시 즉시 로드 실패 에러 발생, 롤백 후 작업 취소
- 데이터 로드 유효성 검사



## VALIDATING DATA LOADS



### PRE-LOAD: COPY INTO - Validation MODE

Allows you to test a data load without executing it

```
[VALIDATION_MODE = RETURN <n> ROWS |  
RETURN_ERRORS | RETURN_ALL_ERRORS]
```



### POST-LOAD: Table Function - VALIDATE

Validates the files loaded in *a past execution* of the **COPY INTO <table>** command

Returns all the errors encountered during the load

```
VALIDATE ( [ <namespace> . ] <table_name> ,  
JOB_ID => { '<query_id>' | _last } )
```

- PRE-LOAD (데이터 로드 전 검사) : COPY INTO 명령에서 VALIDATION\_MODE 파라미터 사용
- POST-LOAD (데이터 로드 후 검사) : VALIDATE 테이블 함수 사용

### • FILE\_FORMAT 생성 예시 (추가 참고)

```
DATA LOADING DEMO + ▾  
▶ Run ☐ All Queries | Saved 1 day ago Context ▾ ...  
16  
17  
18 --Create the file format:  
19 CREATE OR REPLACE FILE FORMAT MY_CSV_FORMAT  
20   TYPE='CSV'  
21   FIELD_DELIMITER='|';  
22  
23 DESCRIBE FILE FORMAT MY_CSV_FORMAT;  
24  
25  
26 --List the stage:  
27 LIST @TRAINING_DB.TRAININGLAB.ED_STAGE/load/lab_files/ pattern='.*region.tbl';  
  
CREATE FILE_FORMAT (Snowsight)
```

- CREATE OR REPLACE FILE\_FORMAT <FILE\_FORMAT 명> TYPE = '타입 (CSV,JSON 일반적인 파일 유형)' FIELD\_DELIMITER="일반적인 FILE 형식에 맞지 않을 경우 지정";  
→ 예시로 생성된 MY\_CSV\_FORMAT은 CSV 타입이지만, DELIMITER 가 | 로 구분되는 파일