```
CREATE DATABASE spotify_db;

CREATE OR REPLACE storage integration s3_init
   TYPE = EXTERNAL_STAGE
   STORAGE_PROVIDER = S3
   ENABLED = TRUE
   STORAGE_AWS_ROLE_ARN = 'arn:aws:iam::241533125232:role/spotify-spark-snowflake-role'
   STORAGE_ALLOWED_LOCATIONS = ('s3://spotify-etl-complete-project')
      COMMENT = 'Creating connection to S3'


DESC integration s3_init;

CREATE OR REPLACE FILE FORMAT csv_fileformat
    TYPE = CSV
    FIELD_DELIMITER = ','
    RECORD_DELIMITER = '\n'
    SKIP_HEADER = 1
    NULL_IF = ('NULL','null', '')
    EMPTY_FIELD_AS_NULL = TRUE
    TRIM_SPACE = TRUE
    ERROR_ON_COLUMN_COUNT_MISMATCH = FALSE;

SHOW DATABASES;

CREATE OR REPLACE stage spotify_stage
    URL = 's3://spotify-etl-complete-project/transformed_data/'
    STORAGE_INTEGRATION = s3_init
    FILE_FORMAT = csv_fileformat;


LIST @spotify_stage/songs;


CREATE OR REPLACE TABLE tbl_album(
        album_id STRING,
        name STRING,
        release_date DATE,
        total_tracks INT,
        url STRING
);


CREATE OR REPLACE TABLE tbl_artists(
        artist_id STRING,
        name STRING,
        url STRING
);


CREATE OR REPLACE TABLE tbl_songs(
        song_id STRING,
        song_name STRING,
        duration_ms INT,
        url STRING,
        popularity INT,
        song_added DATE,
        album_id STRING,
        artist_id STRING
);

COPY INTO tbl_songs
FROM @spotify_stage/songs/songs_transformed_2025-02-19/run-1739998598433-part-r-00000;

COPY INTO tbl_artists
FROM @spotify_stage/artist/artist_transformed_2025-02-19/run-1739999596568-part-r-00008;


COPY INTO tbl_album
FROM @spotify_stage/album/album_transformed_2025-02-19/run-1739998589964-part-r-00001;







COPY INTO tbl_songs
FROM @spotify_stage/songs/songs_transformed_2025-02-19/run-1739998598433-part-r-00000
FILE_FORMAT = (TYPE = CSV, FIELD_DELIMITER = ',', RECORD_DELIMITER = '\n', SKIP_HEADER
= 1, FIELD_OPTIONALLY_ENCLOSED_BY='"');

SELECT * FROM tbl_songs;


LIST @spotify_stage;

------ snow pipe ------
USE SCHEMA PUBLIC;

CREATE OR REPLACE PIPE spotify_db.pipe.tbl_songs_pipe
AUTO_INGEST = TRUE
AS
COPY INTO public.tbl_songs
FROM @spotify_db.public.spotify_stage/songs;

CREATE OR REPLACE PIPE spotify_db.pipe.tbl_artists_pipe
AUTO_INGEST = TRUE
AS
COPY INTO public.tbl_artists
FROM @spotify_db.public.spotify_stage/artist;

CREATE OR REPLACE PIPE spotify_db.pipe.tbl_album_pipe
AUTO_INGEST = TRUE
AS
```

```
COPY INTO public.tbl_album
FROM @spotify_db.public.spotify_stage/album;


DESC pipe tbl_songs_pipe;

DESC pipe tbl_album_pipe;

SELECT COUNT(*) FROM tbl_songs;

ALTER PIPE PIPE.TBL_SONGS_PIPE REFRESH;
ALTER PIPE PIPE.TBL_ARTISTS_PIPE REFRESH;
ALTER PIPE PIPE.TBL_ALBUM_PIPE REFRESH;


SELECT SYSTEM$PIPE_STATUS('pipe.tbl_aritists_pipe');

LIST @spotify_stage;

DESC pipe tbl_artists_pipe;

SELECT COUNT(*) FROM tbl_artists;
SELECT COUNT(*) FROM tbl_album;
SELECT COUNT(*) FROM tbl_songs;

SHOW PIPES;

SELECT SYSTEM$PIPE_STATUS('spotify_db.pipe.tbl_songs_pipe');
SELECT SYSTEM$PIPE_STATUS('spotify_db.pipe.tbl_artists_pipe');
SELECT SYSTEM$PIPE_STATUS('spotify_db.pipe.tbl_album_pipe');


LIST @spotify_stage


SELECT *
FROM TABLE(
    INFORMATION_SCHEMA.COPY_HISTORY(
        TABLE_NAME => 'PUBLIC.TBL_ALBUM',
        START_TIME => DATEADD('HOUR', -1, CURRENT_TIMESTAMP)
    )
)
ORDER BY LAST_LOAD_TIME DESC;


SELECT SYSTEM$PIPE_STATUS('spotify_db.pipe.tbl_album_pipe');
SELECT SYSTEM$PIPE_STATUS('spotify_db.pipe.tbl_songs_pipe');
SELECT SYSTEM$PIPE_STATUS('spotify_db.pipe.tbl_artists_pipe');

SELECT *
FROM TABLE(INFORMATION_SCHEMA.COPY_HISTORY)
WHERE TABLE_NAME = 'PUBLIC.TBL_ALBUM'
```

```
ORDER BY LAST_LOAD_TIME DESC;


SELECT *
FROM TABLE(
    INFORMATION_SCHEMA.COPY_HISTORY(
        START_TIME => DATEADD('HOUR', -1, CURRENT_TIMESTAMP)
    )
)
ORDER BY LAST_LOAD_TIME DESC;
```