1 Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної

техніки Кафедра інформатики та програмної

інженерії

Звіт

з лабораторної роботи №1 з дисципліни
«Аналіз даних в інформаційних системах»

«Створення сховища даних»

Варіант 7

Виконав студент ІП-12 Васильєв Єгор Костянтинович
(шифр, прізвище, ім'я, по батькові)

Перевірив Олійник Юрій Олександрович
(прізвище, ім'я, по батькові)

Київ 2023

Лабораторна робота № 1

Тема: Створення сховища даних

Для виконання лабораторної роботи було обрано 3 джерела відкритих даних на сайті https://www.kaggle.com//. А саме:

1) Інформація про музичні альбоми та їх рейтинг за Metacritic: https://www.kaggle.com/datasets/patkle/metacritic-scores-for-games-movies-tv-and-music
2) Пісні з Spotify та статистика за ними: https://www.kaggle.com/datasets/rodolfofigueroa/spotify-12m-songs
3) Заголовки газети Irish Times, які охоплюють чверть минулого століття: https://www.kaggle.com/datasets/therohk/ireland-historical-news

Таблиця 1 – поля вхідних файлів

SpotifySongs.csv	name	назва треку
	artist_name	ім’я виконавця(ів)
	album_name	назва альбому
	danceability	наскільки трек підходить для танців
	explicit	чи є пісня відвертою
	energy	наскільки насичений і активний трек
	loudness	гучність треку в децибелах
	is_major	чи трек у мажорному режимі або мінорному
	speechiness	відсоток вимовлених слів у треку
	tempo	темп треку, в ударах на хвилину

	duration_ms	тривалість треку у мілісекундах
	release_date	дата релізу треку
	id	id треку
AlbumsScoresMetacritic.csv	name	назва альбому
	album_description	опис альбому
	metascore_rating	оцінка metascore
	artist_name	ім’я виконавця
	metacritic_users_rating	оцінка користувачів metacritic
	release_date	дата релізу альбому
NewsHeadlines.csv	publis_date	дата публікації
	headline_category	категорія заголовку
	headline_text	вміст заголовку

В результаті аналізу початкових даних була спроектована схема stage-зони, яка зображена на рисунку 1. Дана модель відображає таблиці для даних із вхідних джерел.

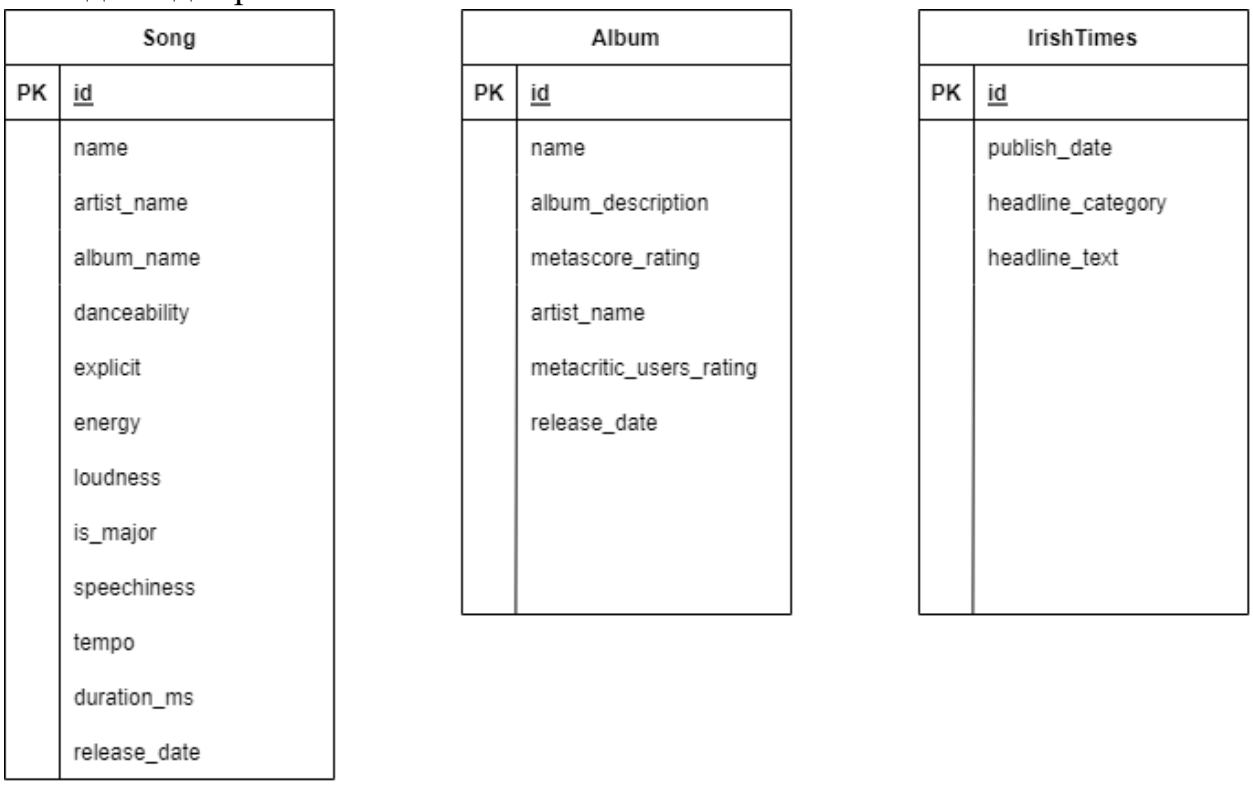

Рисунок 1 – схема stage-зони

Предметною областю лабораторної роботи є музичні альбоми та композиції. У моделі сховища за типом сніжинка спроектовано дві таблиці фактів та шість таблиць вимірів, що показані на рисунку 2.

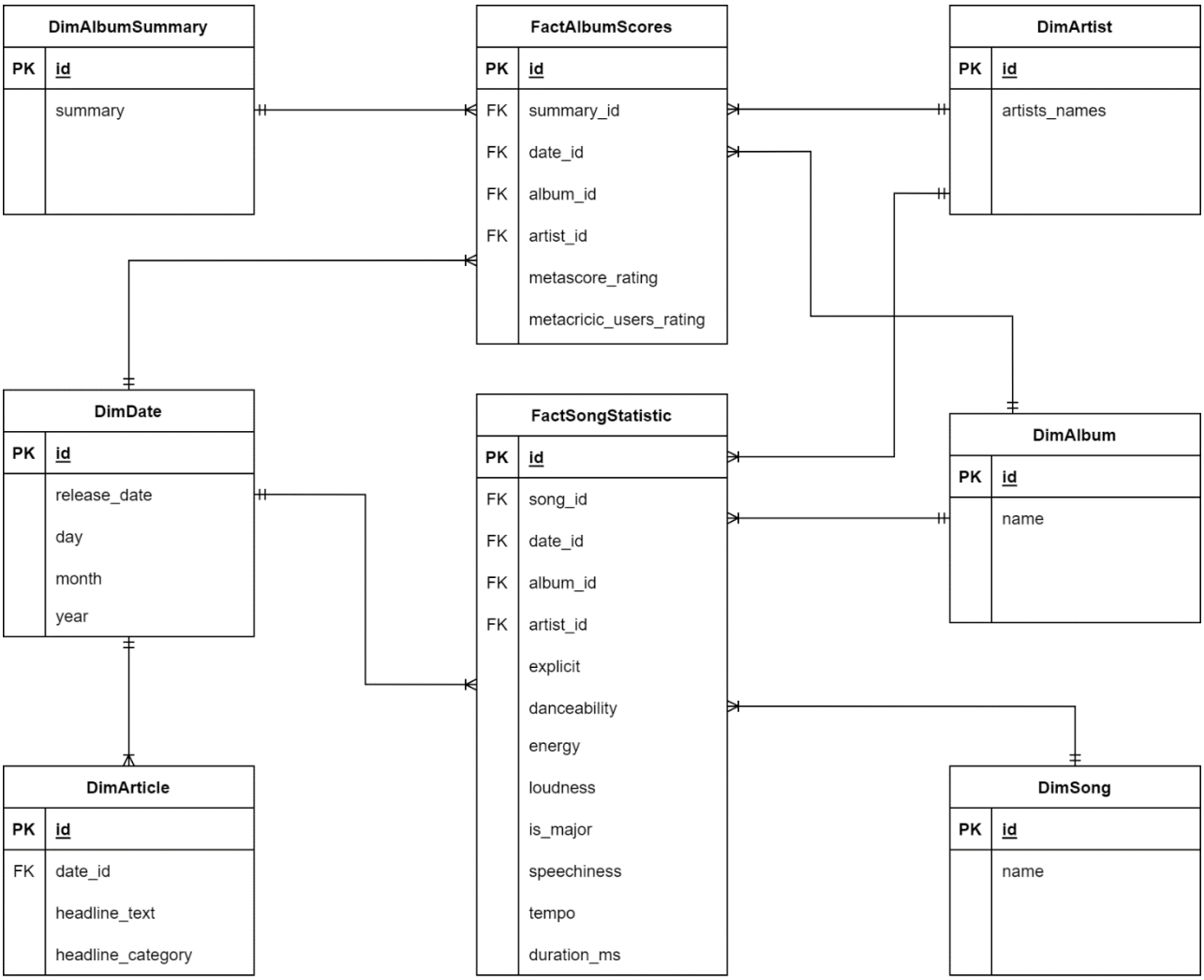

Рисунок 2 – схема сховища даних

Алгоритм завантаження даних у сховище даних

1. Було оброблено вхідні дані:
   1.1. За допомогою методу astype було зведено дані стовпців album_description, name, metacritic_users_rating файлу AlbumsScoresMetacritic до одного типу.
   1.2. За допомогою методу astype було зведено дані стовпців publish_date, headline_text файлу NewsHeadlines до одного типу.
   1.3. За допомогою функції is_not_date було видалено недійсні дані стовпця release_date файлу SpotifySongs.
   1.4. За допомогою функції clean_artists було зведено дані стовпця artists_names файлу SpotifySongs до одного типу.
   1.5. За допомогою функції convert_date було зведено дату у файлах NewsHeadlines, AlbumsScoresMetacritic до одного типу

2. Було завантажено дані до Stage зони:
   2.1. За допомогою функції validate_engine перевірено можливість доступу до бази даних через об’єкт engine бібліотеки sqlalchemy.
   2.2. За допомогою методу to_sql було імпортовано дані до stage зони.
3. Було завантажено дані з Stage зони до основного сховища:
   3.1. За допомогою SQL запитів файлу InsertingValues.sql, наведених у Додатку А було імпортовано дані до сховища.
4. Передбачено можливість завантаження змінених та додаткових даних:
   4.1. За допомогою функції update_summary передбачено перезапис опису альбому в таблиці виміру DimAlbumSummary. У файлі NewSummary.csv представлено два поля: id запису який необхідно змінити та новий опис альбому. Для наочного представлення спрацювання функції update_summary змінимо запис з id=4 та id=19 на «test_description1» і «test_description2» відповідно. Окрім того, в одному рядку файла NewSummary.csv пропустимо значення поля id, для того щоб новий опис просто додався в таблицю.

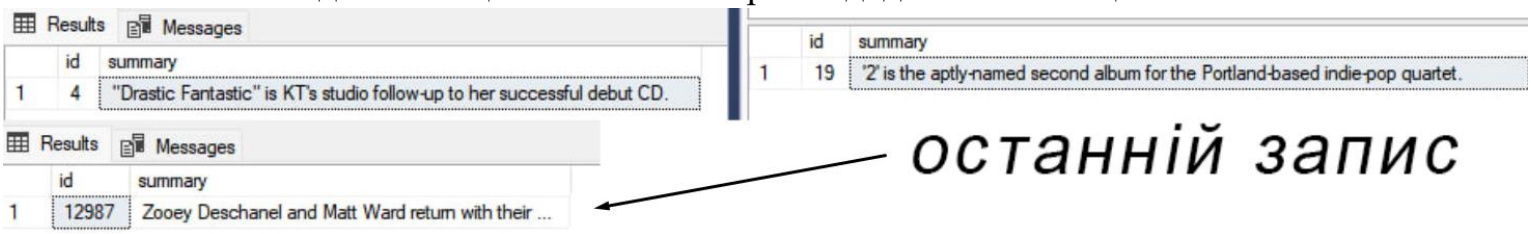

Рисунок 3 – вміст сховища до спрацювання функції update_summary

```
id,new_description
4,test_description1
19,test_description2
,test_description3
```

Рисунок 4 – вміст файлу NewSummary.csv з одним пропуском

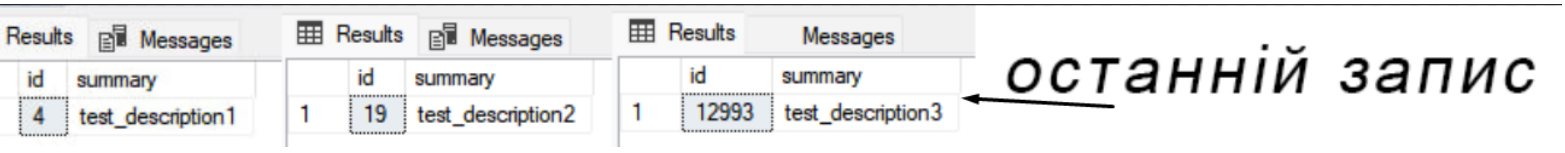

Рисунок 5 – вміст сховища після спрацювання функції update_summary

Висновок: при виконанні даної лабораторної роботи було розглянуто проектування моделі сховища даних за типом сніжинка. Спроектовано модель Stage зони для ETL процесів. Було створено функції для очищення та завантаження даних до Stage зони засобами мови програмування python. Було розроблено SQL запити для внесення даних до основного сховища зі Stage зони. Також було передбачено завантаження змінених та додаткових даних, а саме за допомогою функції update_summary було реалізовано перезапис DimAlbumSummary для зміни опису альбому.

ДОДАТОК А ТЕКСТИ ПРОГРМАНОГО КОДУ

Файл Lab1.py (очищення вхідних даних та завантаження їх у Stage зону)

```
import datetime
import time
import pandas as pd
import numpy as np
import ast
from dateutil.parser import parse
import sqlalchemy


def clean_artists(artists_names):
    artists_names = artists_names[1:-1]
    artists_names = ast.literal_eval(artists_names)
    if isinstance(artists_names, str):
        return artists_names
    else:
        return ', '.join(artists_names)


def convert_date(date):
    return str(parse(date).strftime('%Y-%m-%d'))


def is_not_date(string):
    try:
        datetime.datetime.strptime(string, '%Y-%m-%d')
        return False
    except ValueError:
        return True


def validate_engine():
    try:
        with engine.connect() as con:
            con.execute(sqlalchemy.text("SELECT 1"))
    except Exception as e:
        print(f'Engine invalid: {str(e)}')


initial_df1 = pd.read_csv(r'F:\Egor\Уроки\Аналіз
даних\Лаб1\SpotifySongs.csv',
                          usecols=[0, 1, 2, 4, 8, 9, 10, 12, 13, 14, 19, 20,
23])
initial_df2 = pd.read_csv(r'F:\Egor\Уроки\Аналіз
даних\Лаб1\AlbumsScoresMetacritic.csv', usecols=[1, 2, 3, 5, 6, 7])
initial_df3 = pd.read_csv(r'F:\Egor\Уроки\Аналіз
даних\Лаб1\NewsHeadlines.csv')

spotify_songs = initial_df1.copy()
album_scores = initial_df2.copy()
news_headlines = initial_df3.copy()
spotify_songs.rename(columns={'album': 'album_name', 'artists':
'artists_names', 'mode': 'is_major'}, inplace=True)
album_scores.rename(
    columns={'artist': 'artist_name', 'metascore': 'metascore_rating',
'summary': 'album_description', 'title': 'name',
             'user_score': 'metacritic_users_rating'}, inplace=True)

album_scores.replace('tbd', np.nan, inplace=True)
album_scores = album_scores.astype({'name': 'str', 'album_description':
```

```
'str', 'metacritic_users_rating': 'float64'})
news_headlines = news_headlines.astype({'publish_date': 'str',
'headline_text': 'str'})
not_dates = spotify_songs['release_date'].apply(is_not_date)
spotify_songs['artists_names'] =
spotify_songs['artists_names'].apply(clean_artists)
album_scores['release_date'] =
album_scores['release_date'].apply(convert_date)
news_headlines['publish_date'] =
news_headlines['publish_date'].apply(convert_date)
spotify_songs['release_date'] = np.where(not_dates, np.nan,
spotify_songs['release_date'])

server = 'MAIN-DESKTOP'
database = 'DataAnalysisLab1'
driver = 'ODBC Driver 17 for SQL Server'
conn_str = f'mssql://@{server}/{database}?driver={driver}'
engine = sqlalchemy.create_engine(conn_str)

validate_engine()
start = time.time()
album_scores.to_sql(name='Album', con=engine, if_exists='append',
index=False)
news_headlines.to_sql(name='IrishTimes', con=engine, if_exists='append',
index=False)
spotify_songs.to_sql(name='Song', con=engine, if_exists='append',
index=False)
print("Total data import time: " + str(time.time() - start))
```

Файл UpdateSummary.py (функція оновлення та додавання даних)

```
import pandas as pd
import sqlalchemy


def validate_engine(curr_engine):
    try:
        with curr_engine.connect() as con:
            con.execute(sqlalchemy.text("SELECT 1"))
    except Exception as e:
        print(f'Engine invalid: {str(e)}')


def update_summary(path, curr_engine):
    df = pd.read_csv(path, header=0, names=['id', 'new_description'])
    with curr_engine.connect() as connection:
        for index, row in df.iterrows():
            row_id = row['id']
            new_desc = row['new_description']
            if pd.isna(row_id):
                try:
                    query = sqlalchemy.text(f"INSERT INTO dbo.DimAlbumSummary
VALUES ('{new_desc}')")
                    connection.execute(query)
                    connection.commit()
                except Exception as e:
                    print(f'Error occurred: {str(e)}')
                    connection.rollback()
            else:
                try:
                    new_desc = new_desc.replace("'", "''")
                    query = sqlalchemy.text(f"UPDATE DimAlbumSummary SET
summary='{new_desc}' WHERE id={int(row_id)}")
                    connection.execute(query)
                    connection.commit()
```

```
                except Exception as e:
                    print(f'Error occurred: {str(e)}')
                    connection.rollback()


path_to_file = r"F:\Egor\Уроки\Аналіз даних\Лаб1\NewSummary.csv"
server = 'MAIN-DESKTOP'
database = 'DataAnalysisLab1'
driver = 'ODBC Driver 17 for SQL Server'
conn_str = f'mssql://@{server}/{database}?driver={driver}'
engine = sqlalchemy.create_engine(conn_str)

validate_engine(engine)
update_summary(path_to_file, engine)
```

Файл CreatingStageTables.sql (створення stage зони)

```
USE DataAnalysisLab1
GO

CREATE TABLE Song
(
id NVARCHAR(24) PRIMARY KEY,
[name] NVARCHAR(570),
album_name NVARCHAR(245),
artists_names NVARCHAR(1175),
[explicit] BIT,
danceability FLOAT,
energy FLOAT,
loudness FLOAT,
is_major BIT,
speechiness FLOAT,
tempo FLOAT,
duration_ms INT,
release_date DATE
)
GO

CREATE TABLE Album
(
id INT PRIMARY KEY IDENTITY,
artist_name NVARCHAR(62),
metascore_rating INT,
release_date DATE,
album_description NVARCHAR(2675),
[name] NVARCHAR(150),
metacritic_users_rating FLOAT
)
GO

CREATE TABLE IrishTimes
(
id INT PRIMARY KEY IDENTITY,
publish_date DATE,
headline_category NVARCHAR(40),
headline_text NVARCHAR(280)
)
GO
```

Файл CreatingDimTables.sql (створення таблиць вимірів основного сховища)

```
USE DataAnalysisLab1
```

```
GO

CREATE TABLE DimAlbumSummary
(
id INT PRIMARY KEY IDENTITY,
summary NVARCHAR(2675)
)
GO

CREATE TABLE DimDate
(
id INT PRIMARY KEY IDENTITY,
release_date DATE,
[day] INT,
[month] INT,
[year] INT
)
GO

CREATE TABLE DimArticle
(
id INT PRIMARY KEY IDENTITY,
date_id INT REFERENCES DimDate(id),
headline_category NVARCHAR(40),
headline_text NVARCHAR(280)
)
GO

CREATE TABLE DimSong
(
id INT PRIMARY KEY IDENTITY,
[name] NVARCHAR(570)
)
GO

CREATE TABLE DimAlbum
(
id INT PRIMARY KEY IDENTITY,
[name] NVARCHAR(245)
)
GO

CREATE TABLE DimArtist
(
id INT PRIMARY KEY IDENTITY,
artists_names NVARCHAR(1175)
)
GO
```

Файл CreatingTactTables.sql (створення таблиць фактів основного сховища)

```
USE DataAnalysisLab1
GO

CREATE TABLE FactAlbumScores
(
id INT PRIMARY KEY IDENTITY,
summary_id INT REFERENCES DimAlbumSummary(id),
date_id INT REFERENCES DimDate(id),
album_id INT REFERENCES DimAlbum(id),
artist_id INT REFERENCES DimArtist(id),
metascore_rating INT,
metacritic_users_rating FLOAT
)
GO
```

```
CREATE TABLE FactSongStatistic
(
id INT PRIMARY KEY IDENTITY,
song_id INT REFERENCES DimSong(id),
date_id INT REFERENCES DimDate(id),
album_id INT REFERENCES DimAlbum(id),
artist_id INT REFERENCES DimArtist(id),
[explicit] BIT,
danceability FLOAT,
energy FLOAT,
loudness FLOAT,
is_major BIT,
speechiness FLOAT,
tempo FLOAT,
duration_ms INT,
)
GO
```

Файл InsertingValues.sql (імпорутвання даних до сховища із Stage зони)

```
USE DataAnalysisLab1
GO

INSERT INTO DimDate
SELECT DISTINCT release_date,
DATEPART(day, release_date), DATEPART(month, release_date), DATEPART(year, release_date)
FROM Song
WHERE NOT EXISTS (
  SELECT 1
  FROM DimDate
  WHERE DimDate.release_date = Song.release_date
);
GO

INSERT INTO DimDate
SELECT DISTINCT release_date,
DATEPART(day, release_date), DATEPART(month, release_date), DATEPART(year, release_date)
FROM Album
WHERE NOT EXISTS (
  SELECT 1
  FROM DimDate
  WHERE DimDate.release_date = Album.release_date
);
GO

INSERT INTO DimDate
SELECT DISTINCT publish_date,
DATEPART(day, publish_date), DATEPART(month, publish_date), DATEPART(year, publish_date)
FROM IrishTimes
WHERE NOT EXISTS (
  SELECT 1
  FROM DimDate
  WHERE DimDate.release_date = IrishTimes.publish_date
);
GO

INSERT INTO DimAlbumSummary
SELECT DISTINCT album_description
FROM Album
GO

INSERT INTO DimArticle
SELECT DISTINCT (SELECT id FROM DimDate WHERE release_date = IrishTimes.publish_date),
headline_category, headline_text
```

```
FROM IrishTimes;
GO

SELECT TOP (1000) headline_category, headline_text, release_date FROM DimArticle
JOIN DimDate ON DimDate.id = DimArticle.date_id
GO

INSERT INTO DimSong
SELECT DISTINCT [name]
FROM Song
GO

INSERT INTO DimAlbum
SELECT DISTINCT album_name
FROM Song
GO

INSERT INTO DimAlbum
SELECT DISTINCT [name]
FROM Album
WHERE NOT EXISTS (
  SELECT 1
  FROM DimAlbum
  WHERE DimAlbum.[name] = Album.[name]
)
GO

INSERT INTO DimArtist
SELECT DISTINCT artists_names
FROM Song
GO

INSERT INTO DimArtist
SELECT DISTINCT artist_name
FROM Album
WHERE NOT EXISTS (
  SELECT 1
  FROM DimArtist
  WHERE DimArtist.artists_names = Album.artist_name
)
GO

INSERT INTO FactAlbumScores
SELECT [sum].id, [date].id, album.id, artist.id,
metascore_rating, metacritic_users_rating
FROM Album a
LEFT JOIN DimAlbumSummary [sum] ON a.album_description = [sum].summary
LEFT JOIN DimDate [date] ON a.release_date =  [date].release_date
LEFT JOIN DimAlbum album ON a.[name] =  album.[name]
LEFT JOIN DimArtist artist ON a.artist_name =  artist.artists_names
GO

INSERT INTO FactSongStatistic
SELECT song.id, [date].id, album.id, artist.id,
[explicit], danceability, energy, loudness, is_major,
speechiness, tempo, duration_ms
FROM Song s
LEFT JOIN DimSong song ON s.[name] = song.[name]
LEFT JOIN DimDate [date] ON s.release_date =  [date].release_date
LEFT JOIN DimAlbum album ON s.album_name =  album.[name]
LEFT JOIN DimArtist artist ON s.artists_names =  artist.artists_names
GO

SELECT * FROM FactSongStatistic
GO
```