University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: _2_

Date: _Oct 20, 2023_

Group Number: _34_

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Shrey Gangwar	76979327	c2p3p	shrey.gangwar21@gmail.com
Greg Berezhnov	79484192	v4s7y	greg.berezhnov@gmail.com
Aftahi Ardi	76337671	m8s4f	aia.ardi12@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

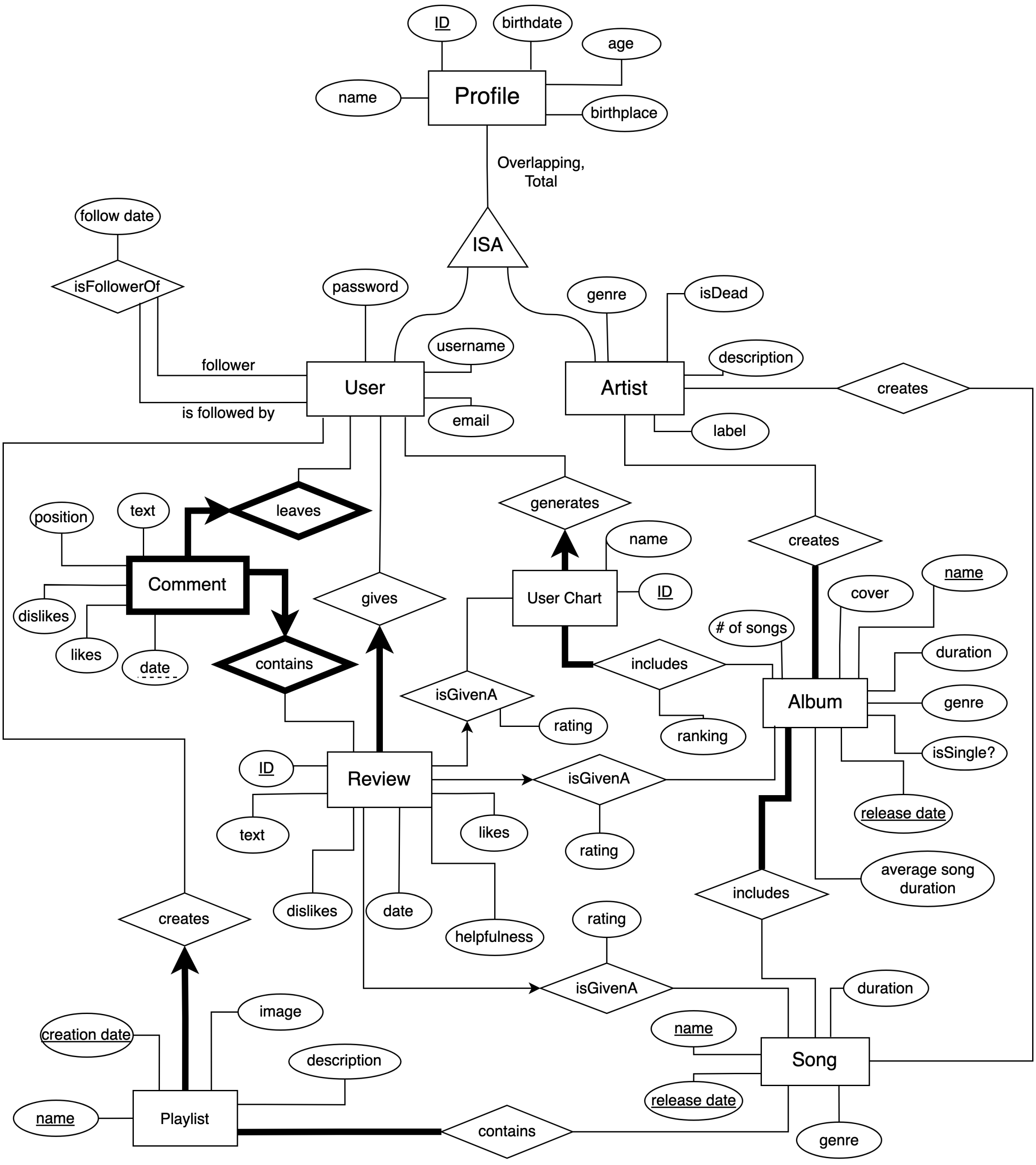
In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Milestone 2

2. Description

Our project aims to create a social platform for music enthusiasts, providing a space for users to share their musical tastes, rate albums and tracks, and curate personalized charts. Users can explore and engage with each other's charts, engaging a community focused on music discovery and critique. Users can also create playlists, allowing the creation of personalized collections of songs.

3. ER Diagram



Changes:

1. Added two more entities in Playlist and Profile, to reach a total of 8. A playlist is made by a user and contains songs.
2. Removed artificial ID keys where they weren't neccessary.
3. Added a more meaningful ISA relationship, with appropriate constraints.
4. Removed the critic ISA relationship since it was not meaningful.
5. Added duration and release date attributes to Song.
6. Song primary key is now the name.
7. Added ranking attribute to User Chart contains Album relationship.
8. Added age and isDead attribute to Artist.
9. Added likes and dislikes to Comment and Review.
10. Added helpfulness attribute to Review.
11. Added position attribute to Comment to track order comments on a Review.

4. Schema

primary key, **foreign key**

- User(pid: INTEGER, name: VARCHAR, birthdate: DATE, birthplace: DATE, username: VARCHAR, password: VARCHAR, email: VARCHAR, age: INTEGER)
  - Candidate Keys: (username)
  - NOT NULL: name, username, password, email
  - UNIQUE: username
- Follows(**followerid**: INTEGER, **followeeid**: INTEGER, follow date: DATE)
  - NOT NULL: follow date
- Artist(pid: INTEGER, name: VARCHAR, birthdate: DATE, birthplace: VARCHAR, genre: VARCHAR, description: VARCHAR, label: VARCHAR, age: INTEGER, isDead: BOOLEAN)
  - NOT NULL: name, genre
  - UNIQUE: (name, birthdate, isDead)
- Review(rid: INTEGER, review date: DATETIME, text: VARCHAR, likes: INTEGER, dislikes: INTEGER, helpfulness: INTEGER, **pid**: INTEGER)
  - NOT NULL: text, likes, dislikes
  - UNIQUE: (review date, likes, dislikes)
- Comment(comment date: DATETIME, text: VARCHAR, likes: INTEGER, dislikes: INTEGER, position: INTEGER, **pid**: INTEGER, **rid**: INTEGER)
  - NOT NULL: text, likes, dislikes
  - UNIQUE: (comment date, likes, dislikes)
- Album(album name: VARCHAR, release date: DATE, isSingle: BOOLEAN, genre: VARCHAR, duration: INTEGER, cover: BLOB, number of songs: INTEGER)
  - Candidate Keys: (album name, duration), (album name, number of songs)
  - NOT NULL: isSingle, genre, duration, cover, number of songs
- Song(song name: VARCHAR, release date: DATE, duration: INTEGER, genre: VARCHAR)
  - Candidate Keys: (song name, duration)
  - NOT NULL: duration, genre
- AlbumSong(**album name**: VARCHAR, **album release date**: DATE, (**song name**: VARCHAR, **song release date**: DATE)
- ArtistAlbum(**pid**: INTEGER, **album name**: VARCHAR, **release date**: DATE)
- ArtistSong(**pid**: INTEGER, **song name**: VARCHAR, **release date**: DATE)
- AlbumReview(**rid**: INTEGER, **album name**: VARCHAR, **release date**: DATE, rating: INTEGER)
  - NOT NULL: rating
- SongReview(**rid**: INTEGER, **song name**: VARCHAR, **release date**: DATE, rating: INTEGER)

  - NOT NULL: rating
- Playlist(playlist name: VARCHAR, creation date: DATE, image: BLOB , description: VARCHAR)
  - NOT NULL: image
- PlaylistSong(**playlist name**: VARCHAR, **creation date**: DATE, **song name**: VARCHAR, **release date**: DATE)
- UserChart(ucid: INTEGER, userchart name: VARCHAR, **pid**: INTEGER)
  - NOT NULL: userchart name
- UserChartAlbum(**ucid**: Integer, **pid**: Integer, **anlbum name**: VARCHAR, **release date**: DATE, ranking: INTEGER)
- UserChartReview(**ucid**: Integer, **pid**: Integer, **rid**: INTEGER)

5. Functional Dependencies

User
- pid → name, birthdate, birthplace, username, password, email, age
- username → email
- username → name
- username, birthdate → age

Follows
- followerid, folloeeid → follow date

Artist
- pid → name, birthdate, birthplace, genre, descrption, label, isDead
- name, birthdate, isDead → age

Comment
- comment date, pid, rid → likes, dislikes, text
- comment date, likes, dislikes → position

Review
- rid, pid → review date, likes, dislikes, text
- review date, likes, dislikes → helpfulness

Album
- album name, release date → isSingle, genre, duration, cover, number of songs

Song
- song name, release date → duration, genre

AlbumReview
- rid, album name, release date → rating

SongReview
- rid, song name, release date → rating

Playlist
- playlist name, creation date → image, description

UserChart
- ucid, pid → userchart name

6. Normalization, BCNF using Decomposition

primary key, **foreign key**

User(pid: INTEGER, name: VARCHAR, birthdate: DATE, birthplace: VARCHAR, username: VARCHAR, password: VARCHAR, email: VARCHAR, age: INTEGER)

- pid → name, birthdate, birthplace, username, password, email, age
- username → email
- username → name
- usernmame, birthdate → age

1. decompose on username → email
   - User1(username: VARCHAR, email: VARCHAR)
   - User2(pid, name, birthdate, birthplace, username, password, age)
     2. decompose on username → name
        - User3(**username**, name)
        - User4(pid, birthdate, birthplace, username, password, age)
          3. decompose on usernme, birthdate → age
             - User5(username, birthdate, age)
             - User6(pid, birthdate, birthplace, username, password)

Final User Tables:

- User1(**username**: VARCHAR, email: VARCHAR)
  - Candidate Keys: N/A
  - NOT NULL: email
  - UNIQUE: N/A
- User3(**username**: VARCHAR, name: VARCHAR)
  - Candidate Keys: N/A
  - NOT NULL: name
  - UNIQUE: N/A
- User5(**username**: VARCHAR, birthdate: DATE, age: INTEGER)
  - Candidate Keys: N/A
  - NOT NULL: N/A
  - UNIQUE: N/A
- User6(pid: INTEGER, birthdate: DATE, birthplace: VARCHAR, username: VARCHAR, password: VARCHAR)
  - Candidate Keys: (username)
  - NOT NULL: username, email
  - UNIQUE: username

Artist(pid: INTEGER, name: VARCHAR, birthdate: VARCHAR, birthplace: VARCHAR, genre: VARCHAR, description: VARCHAR, label: VARCHAR, age: INTEGER, isDead: BOOLEAN)

- pid → name, birthdate, birthplace, genre, descrption, label, isDead
- name, birthdate, isDead → age

1. decompose on name, birthdate, isDead → age
   - Artist1(name, birthdate, isDead, age)
   - Artist2(pid, name, birthdate, birthplace, genre, description, label, isDead)

Final Artist Tables:

- Artist1(**name**: VARCHAR, **birthdate**: DATETIME, **isDead**: BOOLEAN, age: INTEGER)
  - Candidate Keys: N/A
  - NOT NULL: N/A
  - UNIQUE: N/A
- Artist2(pid: INTEGER, name: VARCHAR, birthdate: DATETIME, birthplace: VARCHAR, genre: VARCHAR, description: VARCHAR, label: VARCHAR, isDead: BOOLEAN)
  - Candidate Keys: N/A
  - NOT NULL: name, genre
  - UNIQUE: (name, birthdate, isDead)

Comment(comment date: DATETIME, text: VARCHAR, likes: INTEGER, dislikes: INTEGER, position: INTEGER, **pid**: INTEGER, **rid**: INTEGER)

- comment date, pid, rid → likes, dislikes, text
- comment date, likes, dislikes → position

1. decompose on comment date, likes, dislikes → position
   - Comment1(comment date, likes, dislikes, position)
   - Comment2(comment date, text, likes, dislikes, **pid**, **rid**)

Final Comment Tables:

- Comment1(**comment date**: DATETIME, **likes**: INTEGER, **dislikes**: INTEGER, position: INTEGER)
  - Candidate Keys: N/A
  - NOT NULL: N/A
  - UNIQUE: N/A
- Comment2(comment date: DATETIME, text: VARCHAR, likes: INTEGER, dislikes: INTEGER, **pid**: INTEGER, **rid**: INTEGER)
  - Candidate Keys: N/A
  - NOT NULL: likes, dislikes, text
  - UNIQUE: (comment date, likes, dislikes)

Review(rid: INTEGER, review date: DATETIME, text: VARCHAR, likes: INTEGER, dislikes: INTEGER, helpfulness: INTEGER, **pid**: INTEGER)

- rid → review date, likes, dislikes, text
- likes, dislikes → helpfulness

1. decompose on review date, likes, dislikes → helpfulness
   - Review1(review date: DATETIME, likes, dislikes, helpfulness)
   - Review2(rid, review date, tex, likes, dislikes, **pid**)

Final Review Tables:

- Review1(**review date**: DATETIME, **likes**: INTEGER, **dislikes**: INTEGER, helpfulness: INTEGER)
  - Candidate Keys: N/A
  - NOT NULL: N/A
  - UNIQUE: N/A
- Review2(rid: INTEGER, review date: DATETIME, text: VARCHAR, likes: INTEGER, dislikes: INTEGER, **pid**: INTEGER)
  - Candidate Keys: N/A
  - NOT NULL: likes, dislikes
  - UNIQUE: (review date, likes, dislikes)

7. SQL DDL Statements

```
CREATE TABLE User1 (
    username VARCHAR PRIMARY KEY,
    email VARCHAR NOT NULL,
    FOREIGN KEY (username) REFERENCES User4(username)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE User2 (
    username VARCHAR PRIMARY KEY,
    name VARCHAR NOT NULL,
    FOREIGN KEY (username) REFERENCES User4(username)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE User3 (
    username VARCHAR,
    birthdate DATE,
    age INTEGER,
    PRIMARY KEY (username, birthdate),
    FOREIGN KEY (username) REFERENCES USER4(username)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE User4 (
    pid INTEGER PRIMARY KEY,
    birthdate DATE,
    birthplace VARCHAR,
    username VARCHAR NOT NULL,
    password VARCHAR,
    UNIQUE (username)
);

CREATE TABLE Follows (
    followerid INTEGER,
    followeeid INTEGER,
    follow_date DATE NOT NULL,
    PRIMARY KEY (followerid, followeeid),
    FOREIGN KEY (followerid) REFERENCES User(pid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (followeeid) REFERENCES User(pid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
);

CREATE TABLE Artist1 (
    name VARCHAR,
    birthdate DATE,
    is_dead BOOLEAN,
    age INTEGER
    PRIMARY KEY (name, birthdate, is_dead),
    FOREIGN KEY (name, birthdate, is_dead)
```

```
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Artist2 (
    pid INTEGER PRIMARY KEY,
    name VARCHAR NOT NULL,
    birthdate DATE,
    birthplace VARCHAR,
    genre VARCHAR NOT NULL,
    description VARCHAR,
    label VARCHAR
);

CREATE TABLE Review1 (
    review_date DATETIME,
    likes INTEGER,
    dislikes INTEGER,
    helpfulness INTEGER,
    PRIMARY KEY (review_date, likes, dislikes),
    FOREIGN KEY (review_date, likes, dislikes) REFERENCES Review2(review_date, likes,
dislikes)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Review2 (
    rid INTEGER,
    review_date DATETIME,
    review_text VARCHAR NOT NULL,
    likes INTEGER NOT NULL,
    dislikes INTEGER NOT NULL,
    pid INTEGER,
    UNIQUE (review_date, likes, dislikes)
    PRIMARY KEY (rid, pid),
    FOREIGN KEY (pid) REFERENCES User4(pid)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
);

CREATE TABLE Comment1 (
    comment_date DATETIME,
    likes INTEGER,
    dislikes INTEGER,
    position INTEGER,
    PRIMARY KEY (comment_date, likes, dislikes),
   FOREIGN KEY (comment_date, likes, dislikes) REFERENCES Comment2(comment_date, likes,
dislikes)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Comment2 (
    comment_date DATETIME,
    review_text VARCHAR NOT NULL,
    likes INTEGER NOT NULL,
    dislikes INTEGER NOT NULL,
    pid INTEGER,
    rid INTEGER,
    UNIQUE (comment_date, likes, dislikes)
    PRIMARY KEY (comment_date, pid, rid),
    FOREIGN KEY (pid) REFERENCES User6(pid)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
    FOREIGN KEY (rid) REFERENCES Review(rid)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Album (
    album_name VARCHAR,
    release_date DATE,
    isSingle BOOLEAN NOT NULL,
    genre VARCHAR NOT NULL,
    duration INTEGER NOT NULL,
    cover BLOB NOT NULL,
    number_of_songs INTEGER NOT NULL,
    PRIMARY KEY (album_name, release_date),
    UNIQUE (album_name, duration),
    UNIQUE (album_name, number_of_songs)
);

CREATE TABLE Song (
    song_name VARCHAR,
    release_date DATE,
    duration INTEGER NOT NULL,
    genre VARCHAR NOT NULL,
    PRIMARY KEY (song_name, release_date),
    UNIQUE (song_name, duration)
);

CREATE TABLE AlbumSong (
    album_name VARCHAR,
    album_release_date DATE,
    song_name VARCHAR,
    song_release_date DATE,
    PRIMARY KEY (albumname, album_release_date, song_name, song_release_date),
        FOREIGN  KEY  (album_name,  album_release_date)  REFERENCES  Album(album_name,
release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
   FOREIGN KEY (song_name, song_release_date) REFERENCES Song(song_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
);

CREATE TABLE ArtistAlbum (
    pid INTEGER,
    album_name VARCHAR,
    release_date DATE,
    PRIMARY KEY (pid, album_name, release_date),
    FOREIGN KEY (pid) REFERENCES Artist(pid)
```

```
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (album_name, release_date) REFERENCES Album(album_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE ArtistSong (
    pid INTEGER,
    song_name VARCHAR,
    release_date DATE,
    PRIMARY KEY (pid, song_name, release_date),
    FOREIGN KEY (pid) REFERENCES Artist(pid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (song_name, release_date) REFERENCES Song(song_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE AlbumReview (
    rid INTEGER,
    album_name VARCHAR,
    release_date DATE,
    rating INTEGER NOT NULL,
    PRIMARY KEY (rid, album_name, release_date),
    FOREIGN KEY (rid) REFERENCES Review(rid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (album_name, release_date) REFERENCES Album(album_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE SongReview (
    rid INTEGER,
    song_name VARCHAR,
    release_date DATE,
    rating INTEGER NOT NULL,
    PRIMARY KEY (rid, song_name, release_date),
    FOREIGN KEY (rid) REFERENCES Review(rid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (song_name, release_date) REFERENCES Song(song_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Playlist (
    playlist_name VARCHAR,
    creation_date DATE,
    image BLOB NOT NULL,
    description VARCHAR,
    PRIMARY KEY (playlist_name, creation_date)
);
```

```sql
CREATE TABLE PlaylistSong (
    playlist_name VARCHAR,
    creation_date DATE,
    song_name VARCHAR,
    release_date DATE,
    PRIMARY KEY (playlist_name, creation_date, song_name, release_date),
      FOREIGN KEY (playlist_name, creation_date) REFERENCES Playlist(playlist_name,
creation_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (song_name, release_date) REFERENCES Song(song_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE UserChart (
    ucid INTEGER,
    userchart_name VARCHAR NOT NULL,
    pid INTEGER,
    PRIMARY KEY (ucid, pid),
    FOREIGN KEY (pid) REFERENCES User(pid)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE UserChartAlbum (
    ucid INTEGER,
    pid INTEGER,
    album_name VARCHAR,
    release_date DATE,
    ranking INTEGER,
    PRIMARY KEY (ucid, pid, album_name, release_date),
    FOREIGN KEY (ucid, pid) REFERENCES UserChart(ucid, pid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (album_name, release_date) REFERENCES Album(album_name, release_date)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE UserChartReview (
    ucid INTEGER,
    pid INTEGER,
    rid INTEGER,
    PRIMARY KEY (ucid, pid, rid),
    FOREIGN KEY (ucid, pid) REFERENCES UserChart(ucid, pid)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
    FOREIGN KEY (rid) REFERENCES Review(rid)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);
```

8. INSERT Statements

```
INSERT
INTO Artist1(name, birthdate, isDead, age)
VALUES
('Ryo Fukui', '1948-06-01', 1, 67),
('NAV','1989-11-03', 0, 34),
('Drake','1986-10-24', 0, 36),
('Kanye West','1977-06-08', 0, 46),
('Kendrick Lamar','1987-06-17', 0, 36);

INSERT
INTO Artist2(pid, name, birthdate, birthplace, genre, description, label, isDead)
VALUES
(52033688, 'Ryo Fukui', '1948-06-01', 'Biratori, Hokkaido, Japan', 'Jazz', 'Ryo Fukui
was a Japanese jazz pianist, he regularly played at his own "Slowboat" jazz club in
Sapporo. His work has seen a spike in popularity after his death, around the year 2010.
', 'Independent', 1),
(62043498, 'NAV', '1989-11-03', 'Toronto, Canada', 'Hip-Hop', 'Navraj Singh Goraya,
known professionally as Nav, is a Canadian rapper, singer, songwriter, and record
producer.', 'XO', 0),
(32512674, 'Drake', '1986-10-24', 'Toronto, Canada', 'Hip-Hop', 'Aubrey Drake Graham,
known professionally as Drake, is a Canadian rapper, singer, songwriter, and actor. He
first gained major recognition as a rapper following the release of his mixtape "Room
for Improvement" in 2006.', 'OVO Sound', 0),
(89320472, 'Kanye West', '1977-06-08', 'Atlanta, Georgia, USA', 'Hip-Hop', 'Kanye Omari
West is an American rapper, singer, songwriter, record producer, and fashion designer.
He is widely regarded as one of the most influential musicians of his generation.',
'GOOD Music', 0),
(48297392, 'Kendrick Lamar', '1987-06-17', 'Compton, California, USA', 'Hip-Hop',
'Kendrick Lamar Duckworth, known as Kendrick Lamar, is an American rapper, songwriter,
and record producer. He is regarded as one of the most skillful and successful hip-hop
artists of his generation.', 'Top Dawg Entertainment' 0)

INSERT
INTO Song(song name, release_date, duration, genre)
VALUES
('Early Summer', '1976-06-07', '644', 'Jazz'),
('Did You Wrong', '2020-05-08', '179', 'Rap'),
('Know Yourself', '2015-02-13',100 ,'Rap'),
('Through the Wire', '2003-09-30', 100, 'Hip hop/UK R&B'),
('Wesley’s Theory', '2015-03-15', 100,'Hip hop/Rap');

INSERT
INTO Album(album_name, release_date, isSingle, genre, duration, cover, number_of_songs)
VALUES
('Scenery', '1976-02-01', false, 'Jazz', 644, 'BLOB_DATA_HERE', 6),
('Good Intentions', '2020-05-08', false, 'Rap', 179, 'BLOB_DATA_HERE', 12),
('If Youre Reading This Its Too Late', '2015-02-13', false, 'Rap', 299, 'BLOB_DATA_HERE',
17),
('The College Dropout', '2003-09-30', false, 'Hip hop/UK R&B', 226, 'BLOB_DATA_HERE',
15),
('To Pimp a Butterfly', '2015-03-15', false, 'Hip hop/Rap', 275, 'BLOB_DATA_HERE', 16);

INSERT
INTO AlbumSong(album_name, album_release_date, song_name, song_release_date)
VALUES
```

```sql
('Scenery', '1976-02-01', 'Early Summer', '1976-06-07'),
('Good Intentions', '2020-05-08', 'Did You Wrong', '2020-05-08'),
('If Youre Reading This Its Too Late', '2015-02-13', 'Know Yourself', '2015-02-13'),
('The College Dropout', '2003-09-30', 'Through the Wire', '2003-09-30'),
('To Pimp a Butterfly', '2015-03-15', 'Wesley’s Theory', '2015-03-15');

INSERT
INTO ArtistAlbum(pid, album_name, release_date)
VALUES
(52033688, 'Scenery', '1976-02-01'),
(62043498, 'Good Intentions', '2020-05-08'),
(32512674, 'If Youre Reading This Its Too Late', '2015-02-13'),
(89320472, 'The College Dropout', '2003-09-30'),
(48297392, 'To Pimp a Butterfly', '2015-03-15');

INSERT
INTO ArtistSong(pid, song_name, release_date)
VALUES
(52033688, 'Early Summer', '1976-06-07'),
(62043498, 'Did You Wrong', '2020-05-08'),
(32512674, 'Know Yourself', '2015-02-13'),
(89320472, 'Through the Wire', '2003-09-30'),
(48297392, 'Wesley’s Theory', '2015-03-15');

INSERT
INTO Playlist(playlist_name, creation_date, image, description)
VALUES
('Jazz Vibes', '2023-09-20', 'BLOB_DATA_HERE', 'Relax with some smooth jazz tunes.'),
('Rap Essentials', '2023-09-21', 'BLOB_DATA_HERE', 'The hottest rap tracks right now.'),
('Hip Hop Classics', '2023-09-22', 'BLOB_DATA_HERE', 'Take a trip down memory lane with
these hip hop classics.'),
('R&B Soul', '2023-09-23', 'BLOB_DATA_HERE', 'Soothing R&B for the soul.'),
('Top Hits', '2023-09-24', 'BLOB_DATA_HERE', 'The chart-topping hits everyone is
listening to.');

INSERT
INTO PlaylistSong(playlist_name, creation_date, song_name, release_date)
VALUES
('Jazz Vibes', '2023-09-20', 'Early Summer', '1976-06-07'),
('Rap Essentials', '2023-09-21', 'Did You Wrong', '2020-05-08'),
('Hip Hop Classics', '2023-09-22', 'Through the Wire', '2003-09-30'),
('R&B Soul', '2023-09-23', 'Know Yourself', '2015-02-13'),
('Top Hits', '2023-09-24', 'Wesley’s Theory', '2015-03-15');

INSERT
INTO AlbumReview(rid, album_name, release_date, rating)
VALUES
(12312312, 'Scenery', '1976-02-01', 5),
(23423423, 'Good Intentions', '2020-05-08', 4),
(34534534, 'If You''re Reading This It''s Too Late', '2015-02-13', 5),
(45645645, 'The College Dropout', '2003-09-30', 5),
(56756756, 'To Pimp a Butterfly', '2015-03-15', 5);

INSERT
INTO SongReview(rid, song_name, release_date, rating)
VALUES
```

```
(67867867, 'Early Summer', '1976-06-07', 5),
(67867868, 'Did You Wrong', '2020-05-08', 4),
(67867869, 'Know Yourself', '2015-02-13', 5),
(67867870, 'Through the Wire', '2003-09-30', 5),
(67867871, 'Wesley’s Theory', '2015-03-15', 5);

INSERT
INTO UserChart(ucid, userchart_name, pid)
VALUES
(1, 'My Favorite Albums', 83726415)
(2, 'Brenda''s Top 5 Albums', 41902873),
(3, 'Hunter''s Top 5 Albums', 62519038),
(4, 'James''s Top 5 Albums', 29384756),
(5, 'Howard''s Top 5 Albums', 80172634);

INSERT
INTO UserChartAlbum(ucid, pid, album_name, release_date, ranking)
VALUES
(10626415, 83726415, 'Scenery', '1976-02-01', 1),
(10626415, 83726415, 'Good Intentions', '2020-05-08', 2),
(10626415, 83726415, 'If You''re Reading This It''s Too Late', '2015-02-13', 3),
(10626415, 83726415, 'The College Dropout', '2003-09-30', 4),
(10626415, 83726415, 'To Pimp a Butterfly', '2015-03-15', 5),
(10626414, 41902873, 'Scenery', '1976-02-01', 1),
(10626414, 41902873, 'Good Intentions', '2020-05-08', 2),
(10626414, 41902873, 'If You''re Reading This It''s Too Late', '2015-02-13', 3),
(10626414, 41902873, 'The College Dropout', '2003-09-30', 4),
(10626414, 41902873, 'To Pimp a Butterfly', '2015-03-15', 5),
(10626413, 62519038, 'The College Dropout', '2003-09-30', 1),
(10626413, 62519038, 'To Pimp a Butterfly', '2015-03-15', 2),
(10626413, 62519038, 'Scenery', '1976-02-01', 3),
(10626413, 62519038, 'Good Intentions', '2020-05-08', 4),
(10626413, 62519038, 'If You''re Reading This It''s Too Late', '2015-02-13', 5),
(10626412, 29384756, 'If You''re Reading This It''s Too Late', '2015-02-13', 1),
(10626412, 29384756, 'The College Dropout', '2003-09-30', 2),
(10626412, 29384756, 'To Pimp a Butterfly', '2015-03-15', 3),
(10626412, 29384756, 'Good Intentions', '2020-05-08', 4),
(10626412, 29384756, 'Scenery', '1976-02-01', 5),
(10626411, 80172634, 'To Pimp a Butterfly', '2015-03-15', 1),
(10626411, 80172634, 'The College Dropout', '2003-09-30', 2),
(10626411, 80172634, 'Scenery', '1976-02-01', 3),
(10626411, 80172634, 'If You''re Reading This It''s Too Late', '2015-02-13', 4),
(10626411, 80172634, 'Good Intentions', '2020-05-08', 5);

INSERT
INTO UserChartReview(ucid, pid, rid)
VALUES
(10626415, 83726415, 67867872),
(10626414, 83726415, 67867873),
(10626413, 83726415, 67867874),
(10626412, 83726415, 67867875),
(10626411, 83726415, 67867876);

INSERT
INTO Review1(review_date, likes, dislikes, helpfulness)
VALUES
```

```
('2023-09-11' , 20, 345, 0),
('2023-09-12', 30, 0, 5),
('2023-09-13', 5, 3, 3),
('2023-09-14', 19, 5, 4),
('2023-09-15' , 27, 18, 2)
('2023-09-16' , 25, 10, 5),
('2023-09-17', 35, 0, 10),
('2023-09-18', 15, 7, 7),
('2023-09-19', 29, 6, 9),
('2023-09-20' , 37, 9, 8)
('2023-09-21', 40, 4, 10);

INSERT
INTO Review2(rid, review_date, review_text, likes, dislikes, pid)
VALUES
(12312312, '2023-09-11', 'This is an amazing!', 20, 345, '83726415'),
(23423423, '2023-09-12', 'I really enjoyed.', 30, 0, '41902873'),
(34534534, '2023-09-13', 'Not my cup of tea.', 5, 3, '62519038'),
(45645645, '2023-09-14', 'Decent effort.', 19, 5, '29384756'),
(56756756, '2023-09-15', 'Could have been better.', 27, 18, '80172634')
(67867867, '2023-09-16', 'Absolutely breathtaking!', 25, 10, '83726415'),
(67867868, '2023-09-17', 'This track is fire.', 35, 0, '41902873'),
(67867869, '2023-09-18', 'A decent song but not my taste.', 15, 7, '62519038'),
(67867870, '2023-09-19', 'A classic, never gets old.', 29, 6, '29384756'),
(67867871, '2023-09-20', 'This song speaks volumes.', 37, 9, '80172634'),
(67867872, '2023-09-21', 'This user chart has a great selection of albums!', 40, 4,
83726415),
(67867873, '2023-09-21', 'WOW!', 40, 4, 83726415),
(67867874, '2023-09-22', 'This kind of sucks ngl', 20, 400, 83726415),
(67867875, '2023-09-24', 'Great!', 47, 1, 83726415),
(67867876, '2023-09-26', 'Sucks!', 15, 45, 83726415);

INSERT
INTO User1(username, email)
VALUES
('AngelPretty', 'pretty_angel458@hotmial.com'),
('twocoffeeonemommy', 'brenda.hepburn1968@gmail.com'),
('PORK_MAN', 'zapbran7777@gmail.com'),
('squaresalad', 'diaz.minecraft@gmail.com'),
('tillyfumpy', 'Howard.S.Buggs1945@gmail.com');

INSERT
INTO User2(username, name)
VALUES
('AngelPretty', 'Sofia Cavalry'),
('twocoffeeonemommy', 'Brenda Hepburn'),
('PORK_MAN', 'Hunter McKinnon'),
('squaresalad', 'James Diaz'),
('tillyfumpy', 'Howard Buggs');

INSERT
INTO User3(username, birthdate, age)
VALUES
('AngelPretty', '2003-05-28', 20),
('twocoffeeonemommy', '1968-13-02', 55),
('PORK_MAN', '2014-09-19', 9),
```

```sql
('squaresalad', '2000-06-27', 23),
('tillyfumpy', '1945-12-12', 77);

INSERT
INTO User4(pid, birthdate, birthplace, username, password)
VALUES
(83726415, '2003-05-28', 'Manitoba, Canada', 'AngelPretty', 'makeup1'),
(41902873, '1968-13-02', 'Indiana, USA', 'twocoffeeonemommy', 'ilovemygrandkids123'),
(62519038, '2014-09-19', 'Ontario, Canada','PORK_MAN', 'Fortnite@123'),
(29384756, '2000-06-27', 'Texas, USA', 'squaresalad', 'Egalitarianism'),
(80172634, '1945-12-12', 'New Jersey, USA', 'tillyfumpy', 'ChickenButt');

INSERT
INTO Follows(followerid, followeeid, follower_date)
VALUES
(62519038, 83726415, '2023-05-05'),
(62519038, 41902873, '2023-05-05'),
(62519038, 62519038, '2023-05-05'),
(62519038, 29384756, '2023-05-05'),
(62519038, 80172634, '2023-05-05');

INSERT
INTO Comment1(comment_date, likes, dislikes, position)
VALUES
('2023-10-19 10:00:00', 45, 2, 1),
('2023-10-19 11:00:00', 50, 1, 2),
('2023-10-19 12:30:00', 38, 10, 3),
('2023-10-19 13:45:00', 24, 4, 4),
('2023-10-19 15:00:00', 60, 5, 5);

INSERT
INTO Comment2(comment_date, review_text, likes, dislikes, pid, rid)
VALUES
('2023-10-19 10:00:00', 'Loved this!', 45, 2, 83726415, 12312312),
('2023-10-19 11:00:00', 'Absolutely agree.', 50, 1, 41902873, 23423423),
('2023-10-19 12:30:00', 'I have mixed feelings.', 38, 10, 62519038, 34534534),
('2023-10-19 13:45:00', 'Nice perspective!', 24, 4, 29384756, 45645645),
('2023-10-19 15:00:00', 'Found this insightful.', 60, 5, 80172634, 56756756);
```