SQL Statements

CREATE TABLE USER

(Username VARCHAR(15) NOT NULL,

Fname VARCHAR(15) NOT NULL,

Lname VARCHAR(15) NOT NULL,

Age INT,

CHECK (Age > 0 AND Age <= 120),

Gender VARCHAR(15),

Password VARCHAR(15) NOT NULL,

AdminFlag BOOL,

ClientFlag BOOL,

Date\_Joined DATE NOT NULL,

PRIMARY KEY (Username) );

CREATE TABLE SONG\_RATING

(Rating\_ID INT NOT NULL,

Username VARCHAR(15) NOT NULL,

Originality\_Score INT NOT NULL

CHECK (Originality\_Score <= 10 AND Originality\_Score >= 0),

Lyric\_Score INT NOT NULL

CHECK (Lyric\_Score <= 10 AND Lyric\_Score >= 0),

Vibe\_Score INT NOT NULL

CHECK (Vibe\_Score <= 10 AND Vibe\_Score >= 0),

Instrumental\_Score INT NOT NULL

CHECK (Instrumental\_Score <= 10 AND Instrumental\_Score >= 0),

Song\_ID INT NOT NULL,

Date\_Given DATE NOT NULL,

PRIMARY KEY (Rating\_ID, Username),

FOREIGN KEY (Username) REFERENCES USER(Username)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Song\_ID) REFERENCES SONG(Song\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE ALBUM\_RATING

(Rating\_ID INT NOT NULL,

Username VARCHAR(15) NOT NULL,

Originality\_score INT NOT NULL

CHECK (Originality\_Score <= 10 AND Originality\_Score >= 0),

Lyric\_Score INT NOT NULL

CHECK (Lyric\_Score <= 10 AND Lyric\_Score >= 0),

Vibe\_Score INT NOT NULL

CHECK (Vibe\_Score <= 10 AND Vibe\_Score >= 0),

Instrumental\_Score INT NOT NULL

CHECK (Instrumental\_Score <= 10 AND Instrumental\_Score >= 0),

Album\_flow\_score INT NOT NULL

CHECK (Album\_flow\_score <= 10 AND Album\_flow\_Score >= 0),

Album\_ID INT NOT NULL,

Date\_Given DATE NOT NULL,

PRIMARY KEY (Rating\_ID, Username),

FOREIGN KEY(Username) REFERENCES USER (Username)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY(Album\_ID) REFERENCES ALBUM (ALBUM\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE ALBUM\_REVIEW

(Review\_ID INT NOT NULL,

Username VARCHAR(15) NOT NULL,

Title VARCHAR(50) NOT NULL,

Body VARCHAR(MAX) NOT NULL,

Upvotes INT NOT NULL DEFAULT 0,

Downvotes INT NOT NULL DEFAULT 0,

Album\_ID INT NOT NULL,

Rating\_ID INT NOT NULL,

Date\_Posted DATE NOT NULL,

PRIMARY KEY (Review\_ID, Username),

FOREIGN KEY(Username) REFERENCES USER (Username)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Album\_ID) REFERENCES ALBUM(Album\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Rating\_ID) REFERENCES ALBUM\_RATING(Rating\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE SONG\_REVIEW

(Review\_ID INT NOT NULL,

Username VARCHAR(15) NOT NULL,

Title VARCHAR(50) NOT NULL,

Body VARCHAR(MAX) NOT NULL,

Upvotes INT NOT NULL DEFAULT 0,

Downvotes INT NOT NULL DEFAULT 0,

Song\_ID INT NOT NULL,

Rating\_ID INT NOT NULL,

Date\_Posted DATE NOT NULL,

PRIMARY KEY (Review\_ID, Username),

FOREIGN KEY(Username) REFERENCES USER (Username)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Song\_ID) REFERENCES SONG(Song\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Rating\_ID) REFERENCES SONG\_RATING(Rating\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE COMMENT

(Comment\_ID INT NOT NULL,

Review\_ID INT NOT NULL,

Upvotes INT NOT NULL DEFAULT 0,

Downvotes INT NOT NULL DEFAULT 0,

Body VARCHAR(MAX) NOT NULL,

Username VARCHAR(15) NOT NULL,

Date\_Posted DATE NOT NULL,

PRIMARY KEY (Comment\_ID, Review\_ID),

FOREIGN KEY (Review\_ID) REFERENCES REVIEW(Review\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Username) REFERENCES USER(Username)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE ALBUM

(Album\_ID INT NOT NULL,

Artist\_ID INT NOT NULL,

Name VARCHAR(128) NOT NULL,

Date\_Released DATE NOT NULL,

PRIMARY KEY (Album\_ID, Artist\_ID),

FOREIGN KEY (Artist\_ID) REFERENCES ARTIST(Artist\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE ALBUM\_GENRE

(Album\_ID INT NOT NULL,

Genre VARCHAR(50) NOT NULL,

PRIMARY KEY (Album\_ID, Genre)

FOREIGN KEY (Album\_ID) REFERENCES ALBUM(Album\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE ARTIST

(Artist\_ID INT NOT NULL,

Name VARCHAR(128) NOT NULL,

Age INT,

CHECK (Age > 0 AND Age <= 120),

Gender VARCHAR(15),

Bio VARCHAR(MAX),

PRIMARY KEY (Artist\_ID));

CREATE TABLE ARTIST\_GENRE

(Artist\_ID INT NOT NULL,

Genre VARCHAR(50) NOT NULL,

PRIMARY KEY (Artist\_ID, Genre),

FOREIGN KEY (Artist\_ID) REFERENCES ARTIST (Artist\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE SONG

(Song\_ID INT NOT NULL,

Artist\_ID INT NOT NULL,

Name VARCHAR(128) NOT NULL,

Album\_ID INT,

Date\_Released DATE NOT NULL,

PRIMARY KEY (Song\_ID, Artist\_ID),

FOREIGN KEY (Artist\_ID) REFERENCES ARTIST(Artist\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Album\_ID) REFERENCES ALBUM(Album\_ID)

ON DELETE SET NULL ON UPDATE CASCADE);

CREATE TABLE SONG\_GENRE

(Song\_ID INT NOT NULL,

Genre VARCHAR(50) NOT NULL,

PRIMARY KEY (Song\_ID, Genre),

FOREIGN KEY (Song\_ID) REFERENCES SONG(Song\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE PRODUCER

(Producer\_ID INT NOT NULL,

Name VARCHAR(128) NOT NULL,

Age INT,

CHECK (Age > 0 AND Age <= 120),

Gender VARCHAR(15),

Bio VARCHAR(MAX),

PRIMARY KEY (Producer\_ID));

CREATE TABLE PRODUCER\_GENRE

(Producer\_ID INT NOT NULL,

Genre VARCHAR(50) NOT NULL,

PRIMARY KEY (Producer\_ID, Genre),

FOREIGN KEY (Producer\_ID) REFERENCES PRODUCER(Producer\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE UPVOTES\_DOWNVOTES\_COMMENT

(Username VARCHAR(15) NOT NULL,

Comment\_ID INT NOT NULL,

Vote\_Type BOOL NOT NULL,

PRIMARY KEY (Username, Comment\_ID),

FOREIGN KEY (Username) REFERENCES USER(Username)

ON DELETE CASCADE ON UPDATE CASCADE),

FOREIGN KEY (Comment\_ID) REFERENCES COMMENT(Comment\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE PRODUCES\_THEIR\_ALBUM

(Album ID INT NOT NULL,

Artist\_ID INT NOT NULL,

PRIMARY KEY (Album, Artist\_ID),

FOREIGN KEY (Album\_ID) REFERENCES ALBUM (Album\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Artist\_ID) REFERENCES ARTIST (Artist\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE PRODUCES\_ALBUM

(Producer\_ID INT NOT NULL,

Album\_ID INT NOT NULL,

PRIMARY KEY (Producer\_ID, Album\_ID),

FOREIGN KEY (Producer\_ID) REFERENCES PRODUCER(Producer\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Album\_ID) REFERENCES ALBUM(Album\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE PRODUCES\_SINGLE

(Song\_ID INT NOT NULL,

Producer\_ID INT NOT NULL,

PRIMARY KEY (Song\_ID, Producer\_ID),

FOREIGN KEY (Song\_ID) REFERENCES SONG(Song\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Producer\_ID) REFERENCES PRODUCER(Producer\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE UPVOTES\_DOWNVOTES\_REVIEW

(Username VARCHAR(15) NOT NULL,

Review\_ID INT NOT NULL,

Vote\_Type BOOL NOT NULL,

PRIMARY KEY (Username, Review\_ID),

FOREIGN KEY (Username) REFERENCES USER(Username)

ON DELETE CASCADE ON UPDATE CASCADE),

FOREIGN KEY (Review\_ID) REFERENCES REVIEW(Review\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE PRODUCES\_THEIR\_SINGLE

(Song\_ID INT NOT NULL,

Artist\_ID INT NOT NULL,

PRIMARY KEY (Song\_ID, Artist\_ID)

FOREIGN KEY (Song\_ID) REFERENCES SONG (Song\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (Artist\_ID) REFERENCES ARTIST (Artist\_ID)

ON DELETE CASCADE ON UPDATE CASCADE);

Function: Add Song

Inputs: @Song\_ID, @Name, @Artist\_ID, @Album\_ID

Outputs: None

Query:  
 INSERT INTO SONG  
 VALUES (@Song\_ID, @Name, @Artist\_ID, @Album\_ID);

Function: Remove Song

Inputs: @Song\_ID

Outputs: None

Query:

DELETE FROM SONG

WHERE Song\_ID = @Song\_ID;

Function: Edit Song

Inputs: @Song\_ID, @Name, @Artist\_ID, @Album\_ID

Outputs: None

Query:

UPDATE SONG

SET Name = @Name, Artist\_ID = @Artist\_ID, Album\_ID = @Album\_ID

WHERE (Song\_ID = @Song\_ID);

Function: Search/View Song

Inputs: @Song\_name,

Outputs: Song\_ID, Name, Artist\_ID, Album\_ID

Query:

SELECT Song\_ID, Name, Artist\_ID, Album\_ID

FROM SONG

WHERE Name LIKE ‘%@Song\_name%’

Function: Add Album

Inputs: @Album\_ID, @Name, @Artist\_ID

Outputs: None

Query:  
 INSERT INTO ALBUM

VALUES (@Album\_ID, @Name, @Artist\_ID);

Function: Remove Album

Inputs: @Album\_ID

Outputs: None

Query:

DELETE FROM ALBUM

WHERE Album\_ID = @Album\_ID;

Function: Edit Album

Inputs: @Album\_ID, @Name, @Artist\_ID

Outputs: None

Query:

UPDATE ALBUM

SET Name = @Name, Artist\_ID = @Artist\_ID

WHERE (Album\_ID = @Album\_ID);

Function: Search/View Album

Inputs: @Name

Outputs: Album\_ID, Name, Artist\_ID

Query:  
 SELECT Album\_ID, Name, Artist\_ID

FROM ALBUM

WHERE Name LIKE ‘%@Name%’;

Function: Add Artist

Inputs: @Artist\_ID, @Name, @Age, @Gender, @Bio

Outputs: None

Query:  
 INSERT INTO ALBUM

VALUES (@Album\_ID, @Name, @Age, @Gender, @Bio);

Function: Remove Artist

Inputs: @Artist\_ID

Outputs: None

Query:

DELETE FROM ARTIST

WHERE Artist\_ID = @Artist\_ID;

Function: Edit Artist

Inputs: @Artist\_ID, @Name, @Age, @Gender, @Bio

Outputs: None

Query:

UPDATE ARTIST

SET Name = @Name, Age = @Age, Gender = @Gender, Bio = @Bio WHERE (Artist\_ID = @Artist\_ID);

Function: Search/View Artist

Inputs: @Name

Outputs: Artist\_ID, Name, Age, Gender, Bio

Query:  
 SELECT Artist\_ID, Name, Age, Gender, Bio

FROM ARTIST

WHERE Name LIKE ‘%@Name%’;

Function: Add Producer

Inputs: @Producer\_ID, @Name, @Age, @Gender, @Bio

Outputs: None

Query:  
 INSERT INTO PRODUCER

VALUES (@Producer\_ID, @Name, @Age, @Gender, @Bio);

Function: Remove Producer

Inputs: @Producer\_ID

Outputs: None

Query:

DELETE FROM PRODUCER

WHERE Producer\_ID = @Producer\_ID;

Function: Edit Producer

Inputs: @Producer\_ID, @Name, @Age, @Gender, @Bio

Outputs: None

Query:

UPDATE PRODUCER

SET Name = @Name, Age = @Age, Gender = @Gender, Bio = @Bio WHERE (Producer\_ID = @Producer\_ID);

Function: Search/View Producer

Inputs: @Name

Outputs: Producer\_ID, Name, Age, Gender, Bio

Query:  
 SELECT Producer\_ID, Name, Age, Gender, Bio

FROM PRODUCER

WHERE Name LIKE ‘%@Name%’;

Function: Search/View Profile

Inputs: @Name

Outputs: @Username, @Fname, @Lname  
Query:

SELECT Username, Fname, Lname  
 FROM USER

WHERE Username LIKE ‘%@Name%’

UNION  
 SELECT Username, Fname, Lname  
 FROM USER  
 WHERE Fname LIKE ‘%@Name%’

UNION  
 SELECT Username, Fname, Lname  
 FROM USER

WHERE Lname LIKE ‘%@Name%’;

Function: Edit Profile

Inputs: @Username, @Fname, @Lname, @Age, @Gender, @Password  
Outputs: None  
Query:  
 UPDATE USER  
 SET Fname = @Fname, Lname = @Lname, Age = @Age, Gender = @Gender,   
 Password = @Password  
 WHERE Username = @Username;

Function: Delete Profile

Inputs: @Username  
Outputs: None  
Query:  
 DELETE FROM USER

WHERE Username = @Username;

Function: View Songs

Inputs: @Artist\_ID

Outputs: Song\_ID, Name, Album\_ID

Query:

SELECT S.Song\_ID, S.Name, S.Album\_ID

FROM SONG AS S

WHERE S.Artist\_ID = @Artist\_ID

Function: View Albums

Inputs: @Artist\_ID

Outputs: Album\_ID, Name, Artist\_ID

Query:

SELECT A.Album\_ID, A.Name, A.Artist\_ID

FROM ALBUM AS A

WHERE A.Artist\_ID = @Artist\_ID

Function: View Songs Produced

Inputs: @Producer\_ID

Outputs: Name

Query:

SELECT Name

FROM SONG

WHERE Song\_ID IN (SELECT Song\_ID

FROM PRODUCES\_SINGLE

WHERE Producer\_ID = @Producer\_ID);

Function: View Albums Produced

Inputs: @Producer\_ID

Outputs: Name

Query:

SELECT Name

FROM ALBUM

WHERE Album\_ID IN (SELECT Album\_ID

FROM PRODUCES\_ALBUM

WHERE Producer\_ID = @Producer\_ID);

Function: View Song Ratings

Inputs: @Song\_ID

Outputs: Originality\_Score, Lyric\_Score, Vibe\_Score, Instrumental\_Score

Query:

SELECT Originality\_Score, Lyric\_Score, Vibe\_Score, Instrumental\_Score

FROM SONG\_RATING

WHERE Song\_ID IN (SELECT Song\_ID

FROM SONG

WHERE Song\_ID = @Song\_id);

Function: Give Song Rating

Inputs: @Rating\_ID, @Originality\_Score, @Lyric\_Score, @Vibe\_Score, @Instrumental\_Score, @Username, @Song\_ID

Outputs: none

Query:

INSERT INTO SONG\_RATING

VALUES (@Rating\_ID, @Originality\_Score, @Lyric\_Score, @Vibe\_Score, @Instrumental\_Score, @Username, @Song\_ID);

Function: Edit Song Rating

Inputs: @Rating\_ID, @Originality\_Score, @Lyric\_Score, @Vibe\_Score, @Instrumental\_Score

Outputs: None

Query:

UPDATE Song\_Rating

SET Originality\_Score = @Originality\_Score, Lyric\_Score = @Lyric\_Score, Vibe\_Score = @Vibe\_Score, Instrumental\_Score = @Instrumental\_Score

WHERE Rating\_ID = @Rating\_ID;

Function: Delete Song Rating

Inputs: @Rating\_ID

Outputs: None

Query:

DELETE FROM SONG\_RATING

WHERE Rating\_ID = @Rating\_ID;

Function: Write Song Review

Inputs: @Review\_ID, @Title, @Body, @Username, @Album\_ID, @Rating\_ID

Outputs: None

Query:

INSERT INTO SONG\_REVIEW

VALUES (@Review\_ID, @Title, @Body, 0, 0, @Username, @Song\_ID, @Rating\_ID);

Function: View Song Reviews

Inputs: @Song\_ID

Outputs: Title, Body, Upvotes, Downvotes, Username

Query:

SELECT Title, Body, Upvotes, Downvotes, Username

FROM SONG\_REVIEW

WHERE Song\_ID IN (SELECT Song\_ID

FROM SONG

WHERE Song\_ID = @Song\_id);

Function: Edit Song Review

Inputs: @Title, @Body, @Review\_ID

UPDATE SONG\_REVIEW

SET Title = @Title, Body = @Body,

WHERE Review\_ID = @Review ID

Function: Delete Song Review

Inputs: @Review\_ID

Outputs: None

Query:

DELETE FROM SONG\_REVIEW

WHERE Review\_ID = @Review\_ID;

Function: Write Album Review

Inputs: @Review\_ID, @Title, @Body, @Username, @Album\_ID, @Rating\_ID

Outputs: None

Query:

INSERT INTO ALBUM\_REVIEW

VALUES (@Review\_ID, @Title, @Body, 0, 0, @Username, @Album\_ID, @Rating\_ID);

Function: View Album Reviews

Inputs: @Album\_ID

Outputs: Title, Body, Upvotes, Downvotes, Username

Query:

SELECT Title, Body, Upvotes, Downvotes, Username

FROM ALBUM\_REVIEW

WHERE Album\_ID IN (SELECT Album\_ID

FROM ALBUM

WHERE Album\_ID = @Album\_id);

Function: Edit Album Review

Inputs: @Title, @Body, @Review\_ID

UPDATE SONG\_REVIEW

SET Title = @Title, Body = @Body,

WHERE Review\_ID = @Review ID

Function: Delete Album Review

Inputs: @Review\_ID

Outputs: None

Query:

DELETE FROM ALBUM\_REVIEW

WHERE Review\_ID = @Review\_ID;

Function: View Album Ratings

Inputs: @Album\_ID

Outputs: Originality\_Score, Lyric\_Score, Vibe\_Score, Instrumental\_Score

Query:

SELECT Originality\_Score, Lyric\_Score, Vibe\_Score, Instrumental\_Score, Album\_Flow\_Score

FROM ALBUM\_RATING

WHERE Album\_ID IN (SELECT Album\_ID

FROM ALBUM

WHERE Album\_ID = @Album\_id);

Function: Give Album Rating

Inputs: @Rating\_ID, @Originality\_Score, @Lyric\_Score, @Vibe\_Score, @Instrumental\_Score, @Album\_flow\_score, @Username, @Album\_ID

Outputs: none

Query:

INSERT INTO ALBUM\_RATING

VALUES (@Rating\_ID, @Originality\_Score, @Lyric\_Score, @Vibe\_Score, @Instrumental\_Score, @Album\_flow\_score, @Username, @Album\_ID);

Function: Edit Album Rating

Inputs: @Rating\_ID, @Originality\_Score, @Lyric\_Score, @Vibe\_Score, @Instrumental\_Score, @Album\_flow\_score

Query:

UPDATE ALBUM\_RATING

SET Originality\_score = @Originality\_Score, Lyric\_Score = @Lyric\_Score, Vibe\_Score = @Vibe\_Score, Instrumental\_Score = @Instrumental\_Score, Album\_flow\_score = @Album\_Flow\_Score

WHERE Rating\_ID = @Rating\_ID

Function: Delete Album Rating

Inputs: @Rating\_ID

Outputs: None

Query:

DELETE FROM ALBUM\_RATING

WHERE Rating\_ID = @Rating\_ID;

Function: View Comments

Inputs: @Review\_ID

Outputs: Comment\_ID, Upvotes, Downvotes, Comment\_Text, Username

Query:

SELECT Comment\_ID, Upvotes, Downvotes, Comment\_Text, Username

FROM COMMENT

WHERE Review\_ID IN (SELECT Review\_ID

FROM ALBUM\_REVIEW

WHERE Review\_ID = @Review\_ID

UNION

SELECT Review\_ID

FROM SONG\_REVIEW

WHERE Review\_ID = @Review\_ID);

Function: Write Comment

Inputs: @Review\_ID, @Comment\_ID, @Comment\_Text, @Username

Outputs: None

Query:

INSERT INTO COMMENT

VALUES (@Comment\_ID, 0, 0, @Comment\_Text, @Review\_ID, @Username);

Function: Delete Comment

Inputs: @Comment\_ID

Outputs: None

Query:

DELETE FROM COMMENT

WHERE Comment\_ID = @Comment\_ID;

Function: Edit Comment

Inputs: @Comment\_ID, @Comment\_Text

Outputs: None

Query:

UPDATE COMMENT

SET Comment\_Text = @Comment\_Text

WHERE Comment\_ID = @Comment\_ID;

Function: Give Review Upvote

Inputs: @Username, @Review\_ID

Outputs: None

Query:

UPDATE REVIEW

SET Upvotes = Upvotes + 1

WHERE Review\_ID = @Review\_ID;

INSERT INTO UPVOTES\_DOWNVOTES\_REVIEW

VALUES (@Username, @Review\_ID, TRUE);

Function: Give Review Downvote

Inputs: @Username, @Review\_ID

Outputs: None

Query:

UPDATE REVIEW

SET Downvotes = Downvotes + 1

WHERE Review\_ID = @Review\_ID;

INSERT INTO UPVOTES\_DOWNVOTES\_REVIEW

VALUES (@Username, @Review\_ID, FALSE);

Function: Remove Review Vote

Inputs: @Username, @Review\_ID

Outputs: None

Query:

UPDATE REVIEW

SET Downvotes = Downvotes - 1

WHERE FALSE = (SELECT Vote\_Type

FROM UPVOTES\_DOWNVOTES\_REVIEW AS U  
 WHERE U.Username = @Username

AND U.Review\_ID = @Review\_ID);

UPDATE REVIEW

SET Upvotes = Upvotes - 1

WHERE TRUE = (SELECT Vote\_Type

FROM UPVOTES\_DOWNVOTES\_REVIEW AS U  
 WHERE U.Username = @Username

AND U.Review\_ID = @Review\_ID);

Function: Give Comment Upvote

Inputs: @Username, @Comment\_ID

Outputs: None

Query:

UPDATE COMMENT

SET Upvotes = Upvotes + 1

WHERE Comment\_ID = @Comment\_ID;

INSERT INTO UPVOTES\_DOWNVOTES\_COMMENT

VALUES (@Username, @Comment\_ID, TRUE);

Function: Give Comment Downvote

Inputs: @Username, @Comment\_ID

Outputs: None

Query:

UPDATE COMMENT

SET Downvotes = Downvotes + 1

WHERE Comment\_ID = @Comment\_ID;

INSERT INTO UPVOTES\_DOWNVOTES\_COMMENT

VALUES (@Username, @Comment\_ID, FALSE);

Function: Remove Comment Vote

Inputs: @Username, @Comment\_ID

Outputs: None

Query:

UPDATE COMMENT

SET Downvotes = Downvotes - 1

WHERE FALSE = (SELECT Vote\_Type

FROM UPVOTES\_DOWNVOTES\_COMMENT AS U  
 WHERE U.Username = @Username

AND U.Review\_ID = @Review\_ID);

UPDATE COMMENT

SET Upvotes = Upvotes - 1

WHERE TRUE = (SELECT Vote\_Type

FROM UPVOTES\_DOWNVOTES\_COMMENT AS U  
 WHERE U.Username = @Username

AND U.Comment\_ID = @Comment\_ID);