University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ____2____

Date: __Oct. 13th, 2024___

Group Number: ____123___

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Sara Zhang	60959509	o9j2b	zhangxiyu100@gmail.com
Koda Tootoosis	47941331	r8j6r	kljtootoosis@gmail.com
Yufei Ren	36267672	b4b3b	xixiryf@126.com

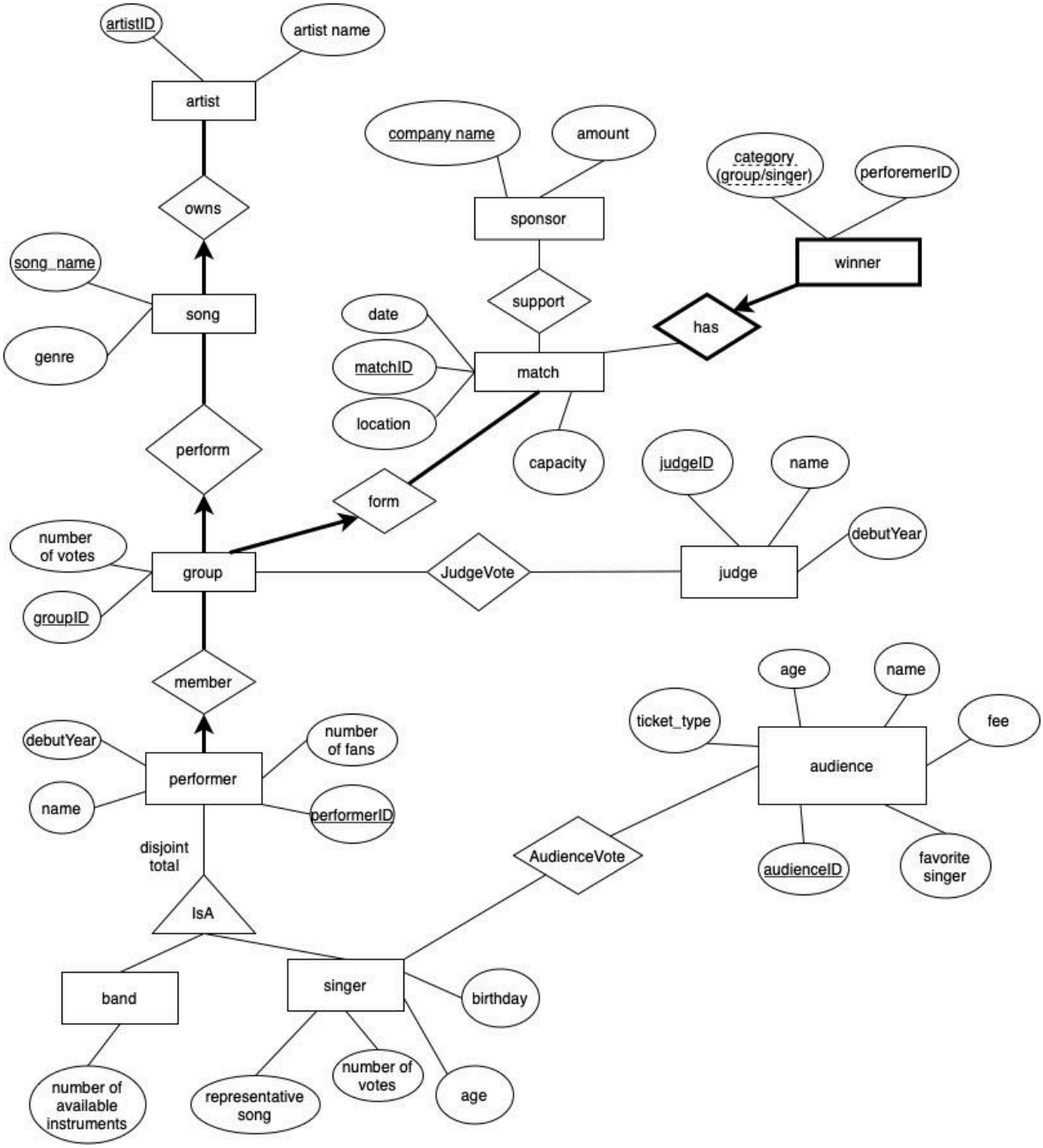
By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address,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

Summary of the project:

The database is to model a voting system of a competition show, involving singers and bands. It has performances that are matches, each match is formed by two groups that contain bands and singers as the performers. The groups then perform a song from the songs in the database. Judges vote on which groups they think are the best ones, and the audience votes on individual singers that they like.

ER Diagram:



Schema/Table definitions & keys

Underlined PK, **Bold FK**

Artist(artistID: integer, artist_name: VARCHAR)
Performer(performerID: integer, performer_name: VARCHAR, debut_year: integer, num_fans: integer, **group-groupID**: Integer NOT NULL on delete SET DEFAULT, on update CASCADE)
Band(num_avail_instr: integer, **performerID**: Integer)
Singer(**performerID**: integer, repre_song: VARCHAR, num_votes: integer, birth_date: Date, age: integer)
Song(song_name: VARCHAR, genre: VARCHAR, **artist-artistID**: integer NOT NULL on delete SET DEFAULT, on update CASCADE)
Sponsor(company_name: VARCHAR, amount: integer)
Audience(audienceID: integer, age: integer, name: VARCHAR, favorite_singer: VARCHAR, ticket_type: VARCHAR, fee: integer)
Judge(judgeID: Integer, name: VARCHAR, debut_year: Integer)
Group(groupID: Integer, number_votes: Integer, **match-matchID**: Integer NOT NULL on delete SET DEFAULT, on update CASCADE, **song-song_name**: VARCHAR NOT NULL on delete SET DEFAULT, on update CASCADE)
Match(matchID: Integer, date: Date, location: VARCHAR, capacity: integer)
Winner(category: VARCHAR, performerID: Integer, matchID: integer)

JudgeVote(**judgeID**: Integer on delete SET DEFAULT, on update CASCADE, **groupID**: Integer on delete SET DEFAULT, on update CASCADE)
AudienceVote(**audienceID**: Integer on delete SET DEFAULT, on update CASCADE, **performerID**: Integer on delete SET DEFAULT, on update CASCADE)
Supports(**company_name**: VARCHAR on delete SET DEFAULT, on update CASCADE, **matchID**: integer on delete SET DEFAULT, on update CASCADE)

Note:

- we cannot model some total participation constraints (e.g., artist, song, group, match) in one-to-many relationship on the one side.
- Clarified names of the vote relationships to JudgeVote and AudienceVote
- We changed the attribute of audience, favorite_singers to favortie_singer, because an attribute can't be a list.
- Changed the attribute name to song_name in the entity song
- added capacity to match entity
- Added birth_date (shown as birthday) and age attributes to singer entity
- Added ticket_type and fee attributes to audience entity

Functional Dependencies (FDs)

Primary keys
artistID -> artist_name
performerID -> performer_name, debut_year, num_fans
performerID -> num_avail_instr
performerID -> repre_song, num_votes, birth_date, age
audienceID -> age, name, favorite_singer, ticket_type, fee
judgeID -> name, debutYear
groupID -> numer_votes, match-matchID, song_name
song_name -> artist-artistID, genre
company_name -> amount
matchID -> date, location, capacity
matchID, category -> performerID

Relationships
judgeID -> groupID
groupID -> judgeID
audienceID -> performerID
performerID -> audienceID
company_name -> matchID
matchID -> company_name

Others
date -> location
location -> capacity
birth_date -> age
ticket_type -> fee

Normalization

All tables are in BCNF other than Match, Audience, and Singer because the left-hand sides of the FDs are keys.

1) Normalization for Match: Match(matchID, date, location, capacity)
   0️⃣ FDs:
      matchID -> date, location, capacity
      date -> location
      location -> capacity
   1️⃣ Take the closure:
      matchID+ = {matchID, date, location, capacity}
      date+ = {date, location, capacity}
      location+ = {location, capacity}
   2️⃣ violates BCNF since date and location are not superkeys, there decompose:

matchID date location, capacity
R1(MatchID, date)
R2(date, location, capacity)

3 violates BCNF since location is not a superkey in R2
date location capacity
R3(date, location)
R4(location, capacity)

4 Final answer:
R1(MatchID, date)
R3(date, location)
R4(location, capacity)

2) Normalization for Audience:
Audience(audienceID, age, name, favorite_singer, ticket_type, fee)

0 FDs:
audienceID -> age, name, favorite_singer, ticket_type, fee
ticket_type -> fee

1 Take the closure:
audienceID+ = {audienceID, age, name, favorite_singer, ticket_type, fee}
ticket_type+ = {ticket_type, fee}
Key: audienceID

2 It violates BCNF, since ticket_type is not a superkey of the relationship
audienceID, age, name, favorite_singer ticket_type fee
R1(audienceID, age, name, favorite_singer, ticket_type)
R2(ticket_type, fee)

3 final answer:
R1(audienceID, age, name, favorite_singer, ticket_type)
R2(ticket_type, fee)

3) Normalization for Singer:
Singer(**performerID**: integer, repre_song: VARCHAR, num_votes: integer, birth_date: Date, age: integer)

0 FDs:
performerID -> repre_song, num_votes, birth_date, age
birth_date -> age

1 Take the closure:
performerID+ = {performerID, repre_song, num_votes, birth_date, age}
birth_date+ = {birth_date, age}
Key: performerID

2 birth_date violates BCNF because it is not a superkey, so we decompose:
performerID, repre_song, num_votes, birth_date age
R1(birth_date, age)
R2(performerID, repre_song, num_votes, birth_date)

3 final answer:

R1(birth_date, age)
R2(performerID, repre_song, num_votes, birth_date)

SQL DDL Statements

```
CREATE TABLE Artist (
        artistID        INTEGER,
        artist_name     VARCHAR(20)         NOT NULL,
        PRIMARY KEY (artistID)
)

CREATE TABLE Performer (
        performerID             INTEGER,
        performer_name          VARCHAR(20)         NOT NULL,
        debut_year              INTEGER,
        num_fans                INTEGER,
        groupID                 INTEGER NOT NULL,
        PRIMARY KEY (performerID)
        FOREIGN KEY (groupID) REFERENCES Group(groupID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Band(
        num_avail_instr         INTEGER,
        performerID             INTEGER NOT NULL,
        PRIMARY KEY (performerID),
        FOREIGN KEY (performerID) REFERENCES Performer(performerID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Singer_R1(
        performerID     INTEGER,
        repre_song      VARCHAR(30),
        num_votes       INTEGER,
        birth_date      DATE,
        PRIMARY KEY (performerID),
        FOREIGN KEY (performerID) REFERENCES Performer(performerID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)
```

```
CREATE TABLE Singer_R2(
        birth_date              DATE,
        age                     INTEGER,
        PRIMARY KEY (birth_date),
        FOREIGN KEY (birth_date) REFERENCES Singer_R1(birth_date)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Song(
        song_name    VARCHAR(30),
        genre        VARCHAR(10),
        artist-artistID   INTEGER NOT NULL,
        PRIMARY KEY (song_name),
        FOREIGN KEY(artist-artistID) REFERENCES Artist(artistID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Sponsor
        (company_name       VARCHAR(20),
         Amount             INTEGER,
         PRIMARY KEY (company_name)
)

CREATE TABLE Audience_Info(
        audienceID          INTEGER,
        age                 INTEGER,
        name                VARCHAR(20),
        favorite_singer     VARCHAR(20),
        ticket_type         VARCHAR(10),
        PRIMARY KEY (audienceID)
)

CREATE TABLE Audience_Ticket(
        ticket_type         VARCHAR(10),
        fee                 INTEGER,
        PRIMARY KEY(ticket_type),
        FOREIGN KEY (ticket_type) REFERENCES Audience_Info(ticket_type)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Judge(
```

```
        judgeID             INTEGER,
        name                VARCHAR(20),
        debut_year          INTEGER,
        PRIMARY KEY (judgeID)
)

CREATE TABLE Group(
        groupID             INTEGER,
        number_votes        INTEGER,
        match-matchID       INTEGER NOT NULL,
        song-song_name      VARCHAR(20) NOT NULL,
        PRIMARY KEY(groupID),
        FOREIGN KEY (match-matchID) REFERENCES Match_Date(matchID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE,
        FOREIGN KEY (song-song_name) REFERENCES Song(song_name)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Match_Date (
        matchID     INTEGER,
        date        DATE,
        PRIMARY KEY (matchID)
)

CREATE TABLE Match_Addr (
        date        DATE,
        location    VARCHAR(20),
        PRIMARY KEY (date),
        FOREIGNKEY (date) REFERENCES Match_Date(date)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Match_Cap(
        location    VARCHAR(20),
        capacity    INTEGER,
        PRIMARY KEY (location),
        FOREIGNKEY (location) REFERENCES Match_Addr(location)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)
```

```
CREATE TABLE Match_Winner (
        performerID     INTEGER,
        category        VARCHAR(20),
        matchID         INTEGER,
        PRIMARY KEY (category, matchID),
        FOREIGN KEY (matchID) REFERENCES Match_Date(matchID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE JudgeVote (
        judgeID         INTEGER,
        groupID         INTEGER,
        PRIMARY KEY (judgeID, groupID),
        FOREIGN KEY (judgeID) REFERENCES Judge(judgeID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE,
        FOREIGN KEY (groupID) REFERENCES Group(groupID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE AudienceVote (
        audienceID      INTEGER,
        performerID     INTEGER,
        PRIMARY KEY (audienceID, performerID),
        FOREIGN KEY (audienceID) REFERENCES Audience_Info(audienceID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE,
        FOREIGN KEY (performerID) REFERENCES Performer(performerID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
)

CREATE TABLE Supports(
        company_name        VARCHAR(20),
        matchID             INTEGER,
        PRIMARY KEY (company_name, matchID),
        FOREIGN KEY (company_name) REFERENCES Sponsor(company_name)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE,
        FOREIGN KEY (matchID) REFERENCES Match_Date(matchID)
                ON DELETE SET DEFAULT
                ON UPDATE CASCADE
```

)

INSERT Statements

INSERT INTO Artist VALUES (6000, ‘Adele’);
INSERT INTO Artist VALUES (6001, ‘Taylor Swift’);
INSERT INTO Artist VALUES (6002, ‘Bruno Mars’);
INSERT INTO Artist VALUES (6003, ‘Ed Sheeran’);
INSERT INTO Artist VALUES (6004, ‘Billie Eilish’);

INSERT INTO Performer VALUES (3000, ‘John’, 2020, 2000, 5000);
INSERT INTO Performer VALUES (3001, ‘Amy’, 2000, 3000, 5001);
INSERT INTO Performer VALUES (3002, ‘Max’, 2006, 5000, 5002);
INSERT INTO Performer VALUES (3003, ‘Elsa’, 2017, 10013, 5003);
INSERT INTO Performer VALUES (3004, ‘Tom’, 2023, 6000, 5004);
INSERT INTO Performer VALUES (3100, ‘Band 1’, 2024, 500, 5000);
INSERT INTO Performer VALUES (3101, ‘Band 2’, 2024, 300, 5001);
INSERT INTO Performer VALUES (3102, ‘Band 3’, 2024, 200, 5002);
INSERT INTO Performer VALUES (3103, ‘Band 4’’, 2024, 600, 5003);
INSERT INTO Performer VALUES (3104, ‘Band 5’, 2024, 100, 5004);

INSERT INTO Band VALUES(8, 3100)
INSERT INTO Band VALUES(5, 3101)
INSERT INTO Band VALUES(2, 3102)
INSERT INTO Band VALUES(4, 3103)
INSERT INTO Band VALUES(7, 3104)

INSERT INTO Singer_R1 VALUES(3000, ‘Happy End’, 100, ‘2000-01-30’)
INSERT INTO Singer_R1 VALUES(3001, ‘Blue Sky’, 80, ‘1995-04-22’)
INSERT INTO Singer_R1 VALUES(3002, ‘Listen’, 90,‘1999-08-15’)
INSERT INTO Singer_R1 VALUES(3003, ‘Ocean’’, 70, ‘2001-10-03’)
INSERT INTO Singer_R1 VALUES(3004, ‘Music World’, 120, ‘1998-02-11’)

INSERT INTO Singer_R2 VALUES(‘2000-01-30’, 24)
INSERT INTO Singer_R2 VALUES(‘1995-04-22’, 29)
INSERT INTO Singer_R2 VALUES(’1999-08-15’, 25)
INSERT INTO Singer_R2 VALUES(‘2001-10-03’, 23)
INSERT INTO Singer_R2 VALUES(‘1998-02-11’, 26)

INSERT INTO Song VALUES (‘Rolling in the Deep’, ‘Pop’, 6000);
INSERT INTO Song VALUES (‘Love Story’, ‘Country’, 6001);
INSERT INTO Song VALUES (‘The Lazy Song’, ‘Reggae’, 6002);
INSERT INTO Song VALUES (‘Shape of You’, ‘Pop’, 6003);
INSERT INTO Song VALUES (‘Bad Guy’, ‘Pop’, 6004);

```
INSERT INTO Sponsor VALUES ('Pepsi', 10000)
INSERT INTO Sponsor VALUES ('Aqiyi', 20000)
INSERT INTO Sponsor VALUES ('Youku', 25000)
INSERT INTO Sponsor VALUES ('Tesla', 9000)
INSERT INTO Sponsor VALUES ('BMO', 500000)

INSERT INTO Audience_Info VALUES (1000, 20, ‘Alice’, ‘SingerA’, ‘General’’)
INSERT INTO Audience_Info VALUES (1001, 36, ‘Bob’, ‘Taylor Swift’, ‘VIP’)
INSERT INTO Audience_Info VALUES (1002, 55, ‘Cindy’, ‘SingerC’, ‘Early Bird’)
INSERT INTO Audience_Info VALUES (1003, 68, ‘Dov’, ‘Ed Sheeran’, ‘Reserved’)
INSERT INTO Audience_Info VALUES (1004, 16, ‘Elyn’, ‘Ed Sheeran’, ‘Child’)

INSERT INTO Audience_Ticket VALUES (‘General’, 50)
INSERT INTO Audience_Ticket VALUES (‘VIP’, 80)
INSERT INTO Audience_Ticket VALUES (‘Early Bird, 40)
INSERT INTO Audience_Ticket VALUES (‘Reserved’, 70)
INSERT INTO Audience_Ticket VALUES (‘Child’, 20)

INSERT INTO Judge VALUES (2000, ‘Judgea’, 2003)
INSERT INTO Judge VALUES (2001, ‘Judgeb’, 1995)
INSERT INTO Judge VALUES (2002, ‘Judgec’, 1999)
INSERT INTO Judge VALUES (2003, ‘Judged’, 2000)
INSERT INTO Judge VALUES (2004, ‘Judgee’, 2005)

INSERT INTO Group VALUES (5000, 750, 4000, ‘Love Story’)
INSERT INTO Group VALUES (5001, 860, 4001, ‘Bad Guy’)
INSERT INTO Group VALUES (5002, 900, 4001, ‘Bad Guy’)
INSERT INTO Group VALUES (5003, 837, 4000, ‘Love Story’)
INSERT INTO Group VALUES (5004, 1012, 4002, ‘Shape of You’)

INSERT INTO Match_Date VALUES (4000, '2023-04-03');
INSERT INTO Match_Date VALUES (4001, '2023-04-03');
INSERT INTO Match_Date VALUES (4002, '2023-04-10');
INSERT INTO Match_Date VALUES (4003, '2023-04-10');
INSERT INTO Match_Date VALUES (4004, '2023-04-17');

INSERT INTO Match_Addr VALUES ('2023-04-03', 'Blue Stadium');
INSERT INTO Match_Addr VALUES ('2023-04-10', 'Red Theatre');
INSERT INTO Match_Addr VALUES ('2023-04-17', 'The Amphitheatre');
INSERT INTO Match_Addr VALUES ('2023-04-24', 'BMO Stadium');
INSERT INTO Match_Addr VALUES ('2023-04-25', 'Madison Square Garden');

INSERT INTO Match_Cap VALUES ('Blue Stadium', 1000);
```

```
INSERT INTO Match_Cap VALUES ('Red Theatre', 1100);
INSERT INTO Match_Cap VALUES ('The Amphitheatre', 900);
INSERT INTO Match_Cap VALUES ('BMO Stadium', 850);
INSERT INTO Match_Cap VALUES ('Madison Square Garden', 1500);

INSERT INTO Match_Winner VALUES (5000, ‘group’, 4000);
INSERT INTO Match_Winner VALUES (5001, 'group', 4001);
INSERT INTO Match_Winner VALUES (3000, 'singer', 4000);
INSERT INTO Match_Winner VALUES (3001, 'singer', 4001);
INSERT INTO Match_Winner VALUES (5004, 'group', 4002);

INSERT INTO JudgeVote VALUES (2000, 5000);
INSERT INTO JudgeVote VALUES (2001, 5001);
INSERT INTO JudgeVote VALUES (2002, 5002);
INSERT INTO JudgeVote VALUES (2003, 5003);
INSERT INTO JudgeVote VALUES (2004, 5004);

INSERT INTO AudienceVote VALUES (1000, 3000);
INSERT INTO AudienceVote VALUES (1001, 3001);
INSERT INTO AudienceVote VALUES (1002, 3002);
INSERT INTO AudienceVote VALUES (1003, 3003);
INSERT INTO AudienceVote VALUES (1004, 3004);

INSERT INTO Supports VALUES ('Pepsi', 4000);
INSERT INTO Supports VALUES ('Aqiyi', 4001);
INSERT INTO Supports VALUES ('Pepsi', 4002);
INSERT INTO Supports VALUES ('Youku', 4003);
INSERT INTO Supports VALUES ('BMO', 4004);
```