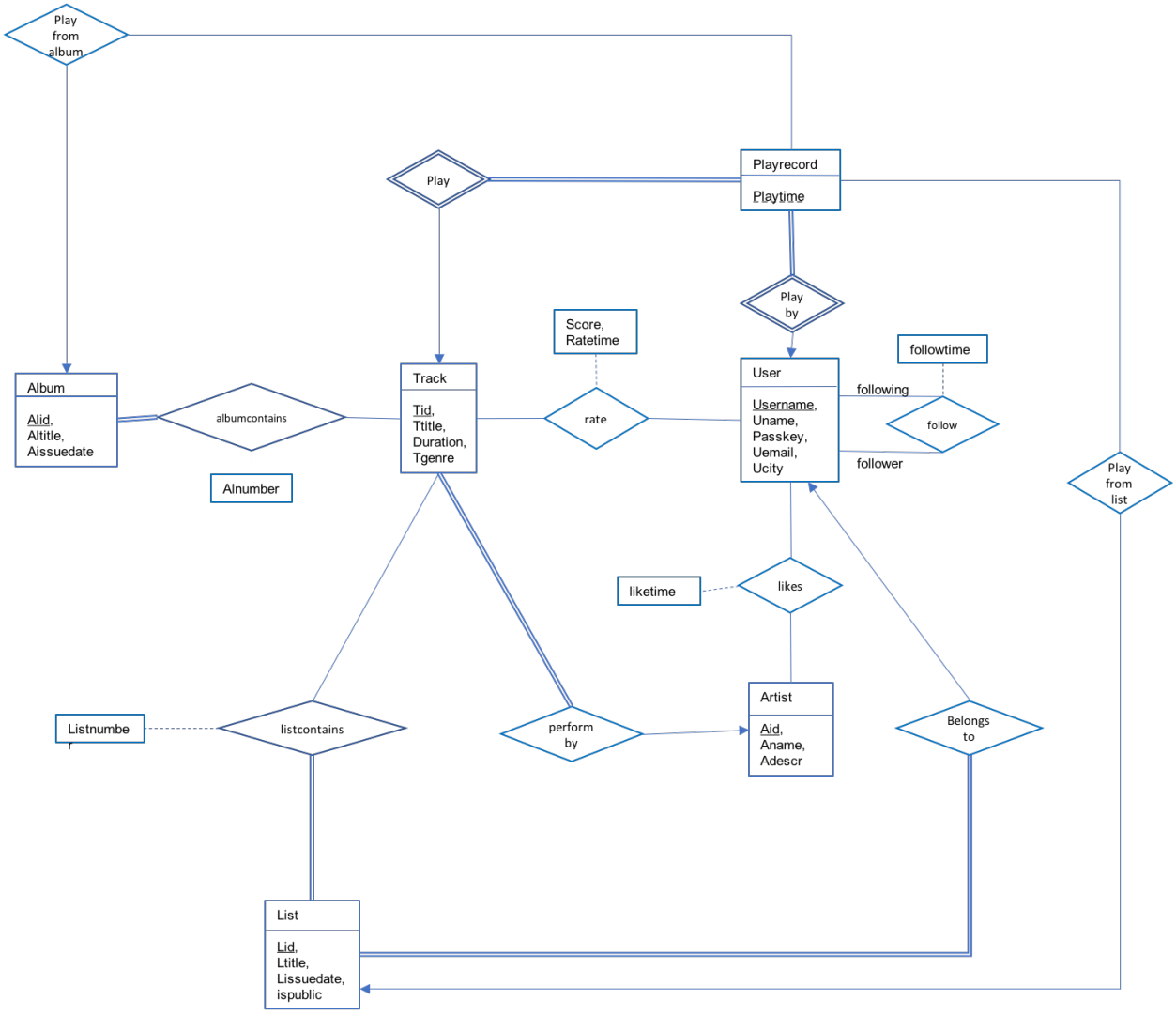
Database – Project 1

ZHIXIN WEN – N14185573
SHUKANG ZHENG - N10908039

1. ER diagram

According to the requirement given, the following ER diagram is designed.



There are 6 entity sets, which are USER, TRACK, ARTIST, ALBUM, LIST and PLAYRECORD. To be noticed, the PLAYRECORD set is a weak entity set which depends on USER and TRACK. The discriminator of PLAYRECORD is *playtime*.

There are 11 relationship sets, which are PLAYBY, PLAY, FOLLOW, RATE, LIKES, ALBUMCONTAINS, LISTCONTAIN, PERFORMBY, PLAYFROMLIST and PLAYFROMALBUM. The following will interpret the relationship sets in detail.

PLAYBY: every PLAYRECORD can be played by only one user but a user can play songs any times. Besides, the participation of PLAYRECORD is total, meaning that every PLAYRECORD must be related to a user.

PLAY: every PLAYRECORD can play only one track, but one track can be played any times. Besides, the participation of PLAYRECORD is total, meaning that every PLAYRECORD must be related to a track.

FOLLOW: a user can follow and be followed by any number of users. The roles in this relationship are *following* and *follower.* FOLLOW has an attribute *followtime*.

RATE: a user can rate any number of tracks and a track can be rated by any number of users. RATE has 2 attributes: *score* and *ratetime*. To be noticed, *score* should be in range of 0 ~ 5.

LIKES: a user can like any number of artists and an artist can be liked by any number of users. LIKES has an attribute *liketime*.

ALBUMCONTAINS: an album can contain any number of tracks and a track can be included in different albums. As an album cannot be empty, the participation of ALBUM is total. As tracks on an album are ordered, ALBUMCONTAINS has an attribute *alnumber* to indicate the track is listed in which place of the album.

LISTCONTAINS: is similar to ALBUMCONTAINS.

PERFORMBY: a track can be performed by only one artist, but an artist can perform any number of tracks. As every track must have an artist, the participation of track is total.

PLAYFROMLIST: the track of a PLAYRECORD can come from only one list, but a list can be played any number of times.

PLAYFROMALBUM: the track of a PLAYRECORD can come from only one album, but a album can be played any number of times.

BELONGSTO: a list belongs to only one user, but a user can have any number of lists. As every list should has an owner, the participation of LIST is total.

2. Reduction to Relational Schemas

(a) Strong entity sets:
User (<u>username</u>, uname, passkey, uemail, ucity)
Track (<u>tid</u>, ttitle, duration, tgenre)
Artist (<u>aid</u>, aname, adescr)
Album (<u>alid</u>, altitle, aissuedate)
List (<u>lid</u>, uid, ltitle, lissuedate, ispublic)

(b) Weak entity sets:
PLAYRECORD depends on both USER AND TRACK, so the primary key is the combination of the primary keys of USER and TRACK and its discriminator.
Playrecord (<u>username</u>, <u>tid</u>, <u>playtime</u>)

(c) Relationship sets:
- Many-to-one: the primary key of the entity set on the “many” side of the relationship set serves as the primary key
Playby (<u>username</u>, <u>tid</u>, <u>playtime</u>)

Play (<u>username</u>, <u>tid</u>, <u>playtime</u>)
Performby (<u>tid</u>, aid)
Playfromlist (<u>username</u>, <u>tid</u>, <u>playtime</u>, lid)
Playfromalbum (<u>username</u>, <u>tid</u>, <u>playtime</u>, alid)

- Many-to-many: the union of the primary-key attributes from the participating entity sets becomes the primary key
Rate (<u>username</u>, <u>tid</u>, score, ratetime)
Likes (<u>username</u>, <u>aid</u>, liketime)
Follow (<u>username1</u>, <u>username2</u>, followtime)
Albumcontains (<u>alid</u>, <u>tid</u>, alnumber)
Listcontains (<u>lid</u>, <u>tid</u>, listnumber)

3. Delete redundancy of schemas

For the many-to-one relationship set in the form AB from entity set A to entity set B. We can combine A and AB to form a single schema consisting of the union of attributes of both schemas. The primary key of the combined schema is the primary key of the entity set into whose schema the relationship set schema was merged. So we have the following combinations.

- PLAYBY, PLAY, PLAYFROMLIST and PLAYFROMALBUM can be combined with PLAYRECORD**.** The resulting PLAYRECORD schema consists of the attributes {*username, tid, playtime, alid, lid*}
- PERFORMBY can be combined with TRACK. The resulting TRACK schema consists of the attributes {*tid, ttitle, duration, tgenre, aid*}
- BELONGSTO can be combined with LIST. The resulting LIST schema consists of the attributes {lid, uid, ltitle, lissuedate, ispublic}

4. Relational table and constraints

The final relational schema is shown below.

User (<u>username</u>, uname, passkey, uemail, ucity)
Primary key: username
Track (<u>tid</u>, ttitle, duration, tgenre, aid)
Primary key: tid
Foreign key:'aid' references aritsit 'aid'
Artist (<u>aid</u>, aname, adescr)
Primary key: aid
Album (<u>alid</u>, altitle, aissuedate)
Primary key:alid
List (<u>lid</u>, uid, ltitle, lissuedate, ispublic)
Primary key:lid
Foreign key:username references user 'username'

Playrecord (<u>username</u>, <u>tid</u>, <u>playtime</u>, alid, lid)
Foreign key:'username' references user 'username'
Foreign key:'tid' references track 'tid'
Foreign key:'alid' references album 'alid'
Foreign key:'lid' references list 'lid'
Rate (<u>username</u>, <u>tid</u>, score, ratetime)
Primary key:username,tid
Foreign key:'username' references user 'username'
Foreign key:'tid' references track 'tid'
Likes (<u>username</u>, <u>aid</u>, liketime)
Primary key:username,aid
Foreign key: 'username' references user 'username'
Foreign key:'aid' reference artist 'aid'
Follow (<u>username1</u>, <u>username2</u>, followtime)
Primary key: username1,username2
Foreign key:'username1' references user 'username'
Foreign key:'username2' references user 'usernmae'
Albumcontains (<u>alid</u>, <u>tid</u>, alnumber)
Primary key: alid,tid
Foreign key:'tid' references track 'tid'
Foreign key:'alid' references album 'alid'
Listcontains (<u>lid</u>, <u>tid</u>, listnumber)
Primary key: lid,tid
Foreign key:'tid' references track 'tid'
Foreign key:'lid' references list 'lid'

5. Queries

(a) Schema creation

```
CREATE TABLE `user` (
  `uname` VARCHAR(45) NOT NULL,
  `username` VARCHAR(45) NOT NULL,
  `passkey` VARCHAR(45) NOT NULL,
  `uemail` VARCHAR(45) NULL,
  `ucity` VARCHAR(45) NULL,
  PRIMARY KEY (`username`));

CREATE TABLE `artist` (
  `aid` INT NOT NULL,
  `aname` VARCHAR(45) NOT NULL,
  `adescr` VARCHAR(200) NOT NULL,
  PRIMARY KEY (`aid`));

CREATE TABLE `track` (
  `tid` INT NOT NULL,
  `ttitle` VARCHAR(45) NOT NULL,
  `duration` TIME NOT NULL,
  `tgenre` VARCHAR(45) NOT NULL,
  `aid` INT NOT NULL,
  PRIMARY KEY (`tid`),
  CONSTRAINT `artist_fk` FOREIGN KEY (`aid`) REFERENCES `artist` (`aid`));
```

```
CREATE TABLE `album` (
  `alid` INT NOT NULL,
  `atitle` VARCHAR(45) NOT NULL,
  `aissuedate` DATE NOT NULL,
  PRIMARY KEY (`alid`));

CREATE TABLE `list` (
  `lid` INT NOT NULL,
  `username` VARCHAR(45) NOT NULL,
  `ltitle` VARCHAR(45) NOT NULL,
  `lissuedate` DATE NOT NULL,
   `ispublic` INT NOT NULL,
  PRIMARY KEY (`lid`),
  CONSTRAINT `user_fk2` FOREIGN KEY (`username`) REFERENCES `user` (`username`));

CREATE TABLE `albumcontains` (
  `tid` INT NOT NULL,
  `alid` INT NOT NULL,
  `alnumber` INT NOT NULL,
  PRIMARY KEY (`alid`, `tid`),
  CONSTRAINT `track_fk` FOREIGN KEY (`tid`) REFERENCES `track` (`tid`),
  CONSTRAINT `album_fk` FOREIGN KEY (`alid`) REFERENCES `album` (`alid`));

CREATE TABLE `listcontains` (
  `tid` INT NOT NULL,
  `lid` INT NOT NULL,
  `listnumber` INT NOT NULL,
  PRIMARY KEY (`lid`, `tid`),
  CONSTRAINT `track_fk2` FOREIGN KEY (`tid`) REFERENCES `track` (`tid`),
  CONSTRAINT `list_fk2` FOREIGN KEY (`lid`) REFERENCES `list` (`lid`));

CREATE TABLE `rate` (
  `username` VARCHAR(45) NOT NULL,
  `tid` INT NOT NULL,
  `ratetime` DATETIME NOT NULL,
  `score` INT NOT NULL CHECK (score >= 1 and score <= 5),
  PRIMARY KEY (`username`,`tid`),
  CONSTRAINT `user_fk3` FOREIGN KEY (`username`) REFERENCES `user` (`username`),
  CONSTRAINT `track_fk3` FOREIGN KEY (`tid`) REFERENCES `track` (`tid`));

CREATE TABLE `follow` (
  `username1` VARCHAR(45) NOT NULL,
  `username2` VARCHAR(45) NOT NULL,
  `followtime` DATETIME NOT NULL,
  PRIMARY KEY (`username1`, `username2`),
  CONSTRAINT `user_fk4` FOREIGN KEY (`username1`) REFERENCES `user` (`username`),
  CONSTRAINT `user_fk5` FOREIGN KEY (`username2`) REFERENCES `user` (`username`));

CREATE TABLE `like` (
  `username` VARCHAR(45) NOT NULL,
  `aid` INT NOT NULL,
  PRIMARY KEY (`username`, `aid`),
  CONSTRAINT `user_fk6` FOREIGN KEY (`username`) REFERENCES `user` (`username`),
  CONSTRAINT `artist_fk2` FOREIGN KEY (`aid`) REFERENCES `artist` (`aid`));
```

```
CREATE TABLE `Playrecord` (
`username` VARCHAR(45) NOT NULL,
`tid` INT NOT NULL,
`playtime` DATETIME NOT NULL,
`alid` INT NULL,
`lid` INT NULL,
PRIMARY KEY (`username`, `tid`, `playtime`),
CONSTRAINT `user_fk7` FOREIGN KEY (`username`) REFERENCES `user` (`username`),
CONSTRAINT `track_fk4` FOREIGN KEY (`tid`) REFERENCES `track` (`tid`),
CONSTRAINT `album_fk2` FOREIGN KEY (`alid`) REFERENCES `album` (`alid`),
CONSTRAINT `list_fk` FOREIGN KEY (`lid`) REFERENCES `list` (`lid`));
```

(b) Queries for question (c)

```
#1
INSERT INTO `user` (uname, username, passkey) VALUES ('zhixin wen', 'zx1347', 'N14185573');
#2
SELECT aid, aname, COUNT(*)
FROM artist NATURAL JOIN playrecord NATURAL JOIN track
GROUP BY aid;
#3
DROP VIEW IF EXISTS `jazztracks`;
CREATE VIEW jazztracks(aid, jnum) as
(SELECT aid, count(*)
FROM track
WHERE tgenre = "Jazz"
GROUP BY aid);

DROP VIEW IF EXISTS `trackcount`;
CREATE VIEW trackcount(aid, tnum) as
(SELECT aid, count(*)
FROM track
GROUP BY aid);

SELECT aid, aname
FROM trackcount NATURAL JOIN jazztracks NATURAL JOIN artist
WHERE jnum >= 0.5 * tnum;
#4
INSERT INTO `rate` (username, tid, ratetime, score) VALUES ('zx1347', 31, NOW(), 4);
#5
SELECT lid, ltitle
FROM user, follow, list
WHERE user.username = follow.username1 and follow.username2 = list.username and user.uname = "Jennifer Lawrence";
#6
SELECT tid, ttitle, aname
FROM track NATURAL JOIN artist
WHERE ttitle like "%just%" OR aname like "%just%";
#7
drop view if exists commonlike;
create view commonlike(aid1, aid2) as (
SELECT like1.aid as aid1, like2.aid
FROM likes as like1, likes as like2
where like1.aid <> like2.aid and like1.username = like2.username and like1.aid < like2.aid
order by aid1);

drop view if exists commoncount;
create view commoncount(aid1, aid2, num) as(
select aid1, aid2, count(*)
from commonlike
group by aid1, aid2);

select aid1, aid2
from commoncount
where num >= 2;
```

6. Sample data

- User

uname	username	passkey	uemail	ucity
Angelina Jolie	AJ	222	Angelina@gmail.com	Los Angeles
Jennifer Lawrence	JL	111	Jennifer@gmail.com	Los Angeles
Julia Roberts	JR	555	Julia@gmail.com	Los Angeles
Marilyn Monroe	MM	444	Marilyn@gmail.com	Chicago
Scarlett Johansson	SJ	333	Scarlett@gmail.com	New York
Shukang Zheng	SZ	000	sz0001@nyu.edu	New York

- Artist

aid	aname	adescr
21	Lady gaga	liberating
22	Britney Spears	Sweet
23	Taylor Swift	Tall
24	Beyonce	icon
25	Justin Bieber	youth

- Track

tid	ttitle	duration	tgenre	aid
31	Bad Romance	00:03:11	Electronic	21
32	Poker Face	00:04:22	Jazz	22
33	Applause	00:05:33	Hip-hop	21
34	Judas	00:03:44	Rock	23
35	Telephone	00:02:59	Electronic	21
36	Just Dance	00:04:26	R&B	22
37	Love Game	00:05:33	Jazz	24
38	The Cure	00:04:18	R&B	25
39	You and I	00:06:49	Dance	24
310	Speechless	00:03:32	Electronic	25

- Album

alid	atitle	aissuedate
41	The Fame	2011-09-11
42	Born This Way	2013-11-05
43	In the Zone	1993-01-05
44	Circus	2014-06-02
45	Blackout	2013-11-05

- List

lid	username	ltitle	lissuedate	ispublic
51	JL	Motivation	2015-01-11	1
52	MM	Focus	2016-09-02	1
53	SJ	Chill	2017-02-14	1
54	AJ	Workout	2017-09-01	1
55	MM	Weekend	2015-01-11	1
56	AJ	Party	2016-09-20	1
57	JR	Romance	2017-05-12	1
58	AJ	Sad	2017-07-29	1
NULL	NULL	NULL	NULL	NULL

- Rate

username	tid	ratetime	score
AJ	33	2015-06-23 17:00:00	4
AJ	35	2016-02-05 16:00:00	1
AJ	37	2017-05-11 16:00:00	5
JL	31	2017-03-08 14:00:00	4
JL	33	2015-04-10 15:00:00	5
JL	34	2015-01-03 12:00:00	2
JL	39	2016-02-05 13:00:00	3
JR	36	2016-12-03 10:03:43	4
JR	39	2017-03-04 12:44:21	3
MM	34	2015-11-09 23:00:12	5
MM	310	2017-10-19 21:00:00	3
SJ	37	2017-08-18 19:30:48	2
SJ	38	2016-07-24 18:00:00	3
SJ	310	2015-04-10 20:49:23	4

- Follow

username1	username2	followtime
AJ	JL	2017-03-12 13:54:33
AJ	MM	2015-04-13 23:09:00
JL	AJ	2015-01-10 10:30:53
JL	SJ	2016-02-11 12:30:08
JR	JL	2015-07-16 23:00:09
MM	JR	2017-06-15 19:00:32
SJ	JL	2016-05-14 08:34:01

- Like

username	aid	liketime
AJ	22	2015-04-17 21:06:56
AJ	23	2016-05-23 16:24:59
AJ	25	2017-03-23 11:22:34
JL	22	2015-01-11 01:01:01
JL	24	2016-02-14 04:02:32
JL	25	2017-06-05 13:22:21
JR	24	2017-09-29 23:37:45
MM	22	2016-08-09 09:50:35
SJ	21	2015-07-07 08:40:34
SJ	25	2017-06-05 13:22:21
NULL	NULL	NULL

- Albumcontains

tid	alid	alnumber
31	41	1
33	41	2
34	41	3
35	41	4
36	42	1
37	42	2
38	42	3
39	43	1
32	44	1
310	44	2
36	45	1

- Listcontains

tid	lid	listnumber
32	51	1
34	51	2
31	52	1
36	53	1
37	53	2
33	54	1
39	55	1
310	55	2
38	56	1
32	57	1
35	57	2
33	58	1
37	58	2

- Playrecord

username	tid	playtime	alid	lid
AJ	33	2015-04-04 17:03:43	NULL	NULL
AJ	34	2017-06-06 21:34:01	NULL	51
AJ	38	2016-05-05 18:00:53	NULL	NULL
JL	31	2015-01-01 10:00:32	41	NULL
JL	32	2016-02-02 12:00:32	44	NULL
JL	37	2017-03-03 11:34:32	42	NULL
JR	32	2015-04-16 09:43:03	NULL	NULL
JR	33	2015-01-13 17:07:04	NULL	54
JR	34	2016-02-14 18:44:35	NULL	NULL
JR	36	2017-03-15 19:55:22	NULL	53
JR	39	2017-12-12 16:06:04	NULL	NULL
MM	31	2015-10-10 12:45:00	43	NULL
MM	310	2016-11-11 15:55:33	NULL	55
SJ	35	2015-07-07 23:34:55	NULL	NULL
SJ	36	2016-08-08 10:06:49	45	NULL
SJ	38	2017-09-09 20:08:45	NULL	56

7. Tests on sample data

(1) Create a record for a new user account, with a name, login name, and a password

before action:

uname	username	passkey	uemail	ucity
Angelina Jolie	AJ	222	Angelina@gmail.com	Los Angeles
Jennifer Lawrence	JL	111	Jennifer@gmail.com	Los Angeles
Julia Roberts	JR	555	Julia@gmail.com	Los Angeles
Marilyn Monroe	MM	444	Marilyn@gmail.com	Chicago
Scarlett Johansson	SJ	333	Scarlett@gmail.com	New York
Shukang Zheng	SZ	000	sz0001@nyu.edu	New York
NULL	NULL	NULL	NULL	NULL

after action:

uname	username	passkey	uemail	ucity
Angelina Jolie	AJ	222	Angelina@gmail.com	Los Angeles
Jennifer Lawrence	JL	111	Jennifer@gmail.com	Los Angeles
Julia Roberts	JR	555	Julia@gmail.com	Los Angeles
Marilyn Monroe	MM	444	Marilyn@gmail.com	Chicago
Scarlett Johansson	SJ	333	Scarlett@gmail.com	New York
Shukang Zheng	SZ	000	sz0001@nyu.edu	New York
zhixin wen	zx1347	N14185573	NULL	NULL
NULL	NULL	NULL	NULL	NULL

(2) For each artist, list their ID, name, and how many times their tracks have been played by users.

aid	aname	COUNT(*)
21	Lady gaga	5
22	Britney Spears	4
23	Taylor Swift	2
24	Beyonce	2
25	Justin Bieber	3

(3) List all artists that are mainly playing Jazz, meaning that at least half of their tracks are of genre Jazz.

aid	aname
22	Britney Spears
24	Beyonce

(4) Insert a new rating given by a use for a track
Before action:

username	tid	ratetime	score
AJ	33	2015-06-23 17:00:00	4
AJ	35	2016-02-05 16:00:00	1
AJ	37	2017-05-11 16:00:00	5
JL	31	2017-03-08 14:00:00	4
JL	33	2015-04-10 15:00:00	5
JL	34	2015-01-03 12:00:00	2
JL	39	2016-02-05 13:00:00	3
JR	36	2016-12-03 10:03:43	4
JR	39	2017-03-04 12:44:21	3
MM	34	2015-11-09 23:00:12	5
MM	310	2017-10-19 21:00:00	3
SJ	37	2017-08-18 19:30:48	2
SJ	38	2016-07-24 18:00:00	3
SJ	310	2015-04-10 20:49:23	4
NULL	NULL	NULL	NULL

after action:

username	tid	ratetime	score
AJ	33	2015-06-23 17:00:00	4
AJ	35	2016-02-05 16:00:00	1
AJ	37	2017-05-11 16:00:00	5
JL	31	2017-03-08 14:00:00	4
JL	33	2015-04-10 15:00:00	5
JL	34	2015-01-03 12:00:00	2
JL	39	2016-02-05 13:00:00	3
JR	36	2016-12-03 10:03:43	4
JR	39	2017-03-04 12:44:21	3
MM	34	2015-11-09 23:00:12	5
MM	310	2017-10-19 21:00:00	3
SJ	37	2017-08-18 19:30:48	2
SJ	38	2016-07-24 18:00:00	3
SJ	310	2015-04-10 20:49:23	4
zx1347	31	2017-12-02 12:41:56	4
NULL	NULL	NULL	NULL

(5) For a particular user, say “NancyInQueens”, list all playlists that were made by users that she follows.

lid	ltitle
54	Workout
56	Party
58	Sad
53	Chill

(6) List all songs where the track title or artist title matches some set of keywords.

tid	ttitle	aname
36	Just Dance	Britney Spears
38	The Cure	Justin Bieber
310	Speechless	Justin Bieber

(7) Find pairs of related artists, where two artists are related if they have many fans in common. (as the sample data is small in scale, we define the related artists as the artists have at least 2 common fans)

aid1	aid2	common_fans
22	25	2