Design and Create Schema

Group 18: Chengzhen Wu, Huiqi Mao, Ruofan Zhou

March 20, 2014

1 The ER model

In the given project data, we firstly recognize 'area', 'artist' and 'genre' each as three individual entities **Each artist is from at most one area, so it's a many-to-one relation.** Several artists can belong to different genres and one genre can contain several artists. So **the relation between 'artist' and 'genre' is many-to-many.**

Secondly, we think about the relationship among 'release','recording',track' and 'medium'. We imagine a scene to describe these relations. The csv file of 'release' contains the names of releases. They could be stored in the mediums, such as CD, 12" Vinyl and so on. What's more, one release could have several CDs to contain many tracks, or in different medium (I'm not sure about this, but possible). So **the relation between 'release' and 'medium' is one-to-many.** Next, each track in different mediums must correspond to one recording. So **the relation between 'track' and 'recording' is many-to-one. Each 'track' must be in one of 'medium's. So the relation is many-to-one.**

Finally, **the relation between 'artist' and 'track' is many-to-many**. Because many artists could participate together in one track. And many tracks(recordings) could be performed by the same artist. In fact, we can consider 'track' is a relation between 'medium' and 'recording', which means artist participate in this whole relation. In other words, **we aggregate the 'release' , 'recording' and 'medium' as the relation of 'track'.**

We draw our ER model as below:
PS. 'count' in 'genre' could be a view, so we ignore this attribution.

2 SQL commands

```
--ENTITY area
CREATE TABLE  area (
  AID INTEGER,
  Aname CHAR(45),
  Atype CHAR(45),
  PRIMARY KEY (AID));

--ENTITY genre
--we don't need gcount in genre
CREATE TABLE  genre (
  GID INTEGER,
  Gname CHAR(45),
  PRIMARY KEY (GID));

--ENTITY artist
CREATE TABLE  artist (
  arID INTEGER,
  name CHAR(45) NULL,
  type CHAR(45) NULL,
  gender CHAR(45) NULL,
  PRIMARY KEY (arID));

--ENTITY track
CREATE TABLE  track (
  TID INTEGER,
  position INTEGER,
  recording_RID INT NOT NULL,
  artist_ID INT NOT NULL,
  artist_area_AID INT NOT NULL,
  artist_genre_GID INT NOT NULL,
  media_MID INT NOT NULL,
  media_release_REID INT NOT NULL,
  PRIMARY KEY (TID));

--ENTITY recording
CREATE TABLE  recording (
  RID INTEGER,
  Rname CHAR(45),
  Rlength CHAR(45),
  PRIMARY KEY (RID));

--ENTITY release
```

```
CREATE TABLE  release (
  REID INTEGER,
  Relname CHAR(45),
  PRIMARY KEY (REID));

--ENTITY media
CREATE TABLE  media (
  MID INTEGER,
  Mformat CHAR(45),
  release_REID INTEGER,
  PRIMARY KEY (MID));

--RELATIONSHIP artist-area
CREATE TABLE artist_area (
  arID INTEGER NOT NULL,
  AID INTEGER,
  PRIMARY KEY (arID, AID),
  FOREIGN KEY (arID) REFERENCES artist,
  FOREIGN KEY (AID) REFERENCES area);

--RELATIONSHIP artist_genre
CREATE TABLE artist_genre (
  arID INTEGER NOT NULL,
  GID INTEGER NOT NULL,
  PRIMARY KEY (arID, GID),
  FOREIGN KEY (arID) REFERENCES artist,
  FOREIGN KEY (GID) REFERENCES genre);

--RELATIONSHIP media_recording
CREATE TABLE media_recording(
  MID INTEGER,
  REID INTEGER,
  PRIMARY KEY (MID, REID),
  FOREIGN KEY (MID) REFERENCES media,
  FOREIGN KEY (REID) REFERENCES recording);

--RELATIONSHIP artist_track
CREATE TABLE artist_track(
  arID INTEGER NOT NULL,
  TID INTEGER NOT NULL,
  PRIMARY KEY (arID, TID),
  FOREIGN KEY (arID) REFERENCES artist,
  FOREIGN KEY (TID) REFERENCES track);
```

```
--RELATIONSHIP trackR
CREATE TABLE trackR(
   TID INTEGER, RID INTEGER, MID INTEGER,
   PRIMARY KEY (TID, RID, MID),
   FOREIGN KEY (RID) REFERENCES recording,
   FOREIGN KEY (TID) REFERENCES track,
   FOREIGN KEY (MID) REFERENCES media);
```