Paul Asa, Jamie Luck
November 11, 2014
Database Systems
Project Stage 2

SQL Statements:

```
create table Artist(
     artist_name                 varchar(70), not null
     real_name                   varchar(40),
     location                    varchar(30),
     website                     varchar(30),
     bio_text                    varchar(500),
     photo_link                  varchar(30),
     primary key (artist_name))

create table Album(
     release_no                  varchar(10), not null
     album_name                  varchar(70), not null
     download_link               varchar(30),
     no_tracks                   numeric(3,0),
     artist_name                 varchar(70), not null
     description                 varchar(500),
     art_link                    varchar(30),
     primary key (release_#),
     foreign key (artist_name) references Artist
          on delete cascade
          on update cascade)

create table Track(
     album_name                  varchar(70), not null
     track_#                     numeric(3,0),
     title                       varchar(70),
     stream_link                 varchar(30),
     primary key (album_name, track_#)
     foreign key (album_name) references Album
          on delete cascade
          on delete cascade)
```

[Note: 8Bit Peoples uses a string for release numbers (i.e. 8BP-160)]

Paul Asa, Jamie Luck
November 11, 2014
Database Systems
Project Stage 2

Justification

`Artist:`

artist_name cannot be null because it is an essential identifier of an Artist entity. No other attribute provides this functionality.

`Album:`

releaese_no, album_name, and artist_name cannot be null because they are all essential pieces for identifying an Album entity.

`Track:`

album_name cannot be null because it is the only attribute that distinguishes where that Track entity belongs.

Additionally, it is important to cascade on delete as Track depends on Album, and Album depends on Artist.

SQL Queries

Search for artist by name - provided with a string, this query will return all (if any) artists that match that string, fully or partially.

```
select * from Artist where artist_name="$$";
```

Get all albums by artist - provided with a string, this query will return all albums (if any) by the artist of that name.

```
select * from Album where artist_name="$$";
```

Get all tracks on album - provided with a string, this query will return all tracks on any albums of that name.

```
select * from Track where album_name="$$";
```

The above three queries provide the most basic functionality of a record label database. Additionally, the following queries may be added for further functionality:

Get albums based on release_no - provided with a string, this query will return all albums released with a release number equal to, or lower than that release_no.

```
select * from Album where release_# >= $$;
```

Search artists based on locations - provided with a string, this query will return all artists with a location that matches that string.

```
select * from Artist where location="New York City";
```