Cole Hoener

Steven Kearns

Coachella Artists

DBMS1

October 18th, 2021

Report Document

Application Description, Industry, and Client/Customer

With a growing music and concert industry across the world, more and more artists perform at music festivals adding to an already extensive lineup. Many of these artists are less commonly known, except for the headliners, and finding information on artists on a lineup poster can be difficult. Our database will provide data on each artist when a user clicks on a name on the lineup poster, either on the music festivals website or on the digital lineup poster. Our data will be based on the Coachella 2019 (because of COVID in 2020) lineup. Anyone interested in music, and more specifically those attending Coachella or any other music festivals who are curious about the artists in the lineup will benefit from this database schema.

Opportunity/Problem

Imagine going to a music festival of your choice and looking up the lineup for the day you decide to go, but you notice the other artists located at the bottom of the lineup sheet. You start to wonder what kind of songs they play and what kind of genre their music is about. Our information will be based on the Coachella 2019 lineup and would allow music festival goers to get to know the music catalog of each artist. Especially for the ones that have their own genre and are located in the smaller text above the bigger headliners.

Entities and Relationships

The main entity that our database management system is centered around is the Artist. Each artist would have the following attributes: stage name, real name, DOB/age, and gender. An artist is part of an entity Label which only has the attribute of label name. Now artists have Albums, Mixtapes/EPs, and Features. Songs which have the attributes of song title and song length are part of either an Album or Mixtape/EP and Genre, which is also an entity with the attributes of just genre type. Features are only part of a Genre and have the following attributes of song title, original artist, featuring artist, and genre. The entities Albums and Mixtapes/EPs have the same attributes of title, and genre which both are part of a Genre and an Artist. Furthermore, artists have social media accounts so the Social Media entity will have the following attributes of twitter, instagram, and facebook. The last entity is Schedule which keeps track of the when and where an artist is playing during Coachella, it has the attributes: date, time, artist, and location. In total that would give us nine entities.

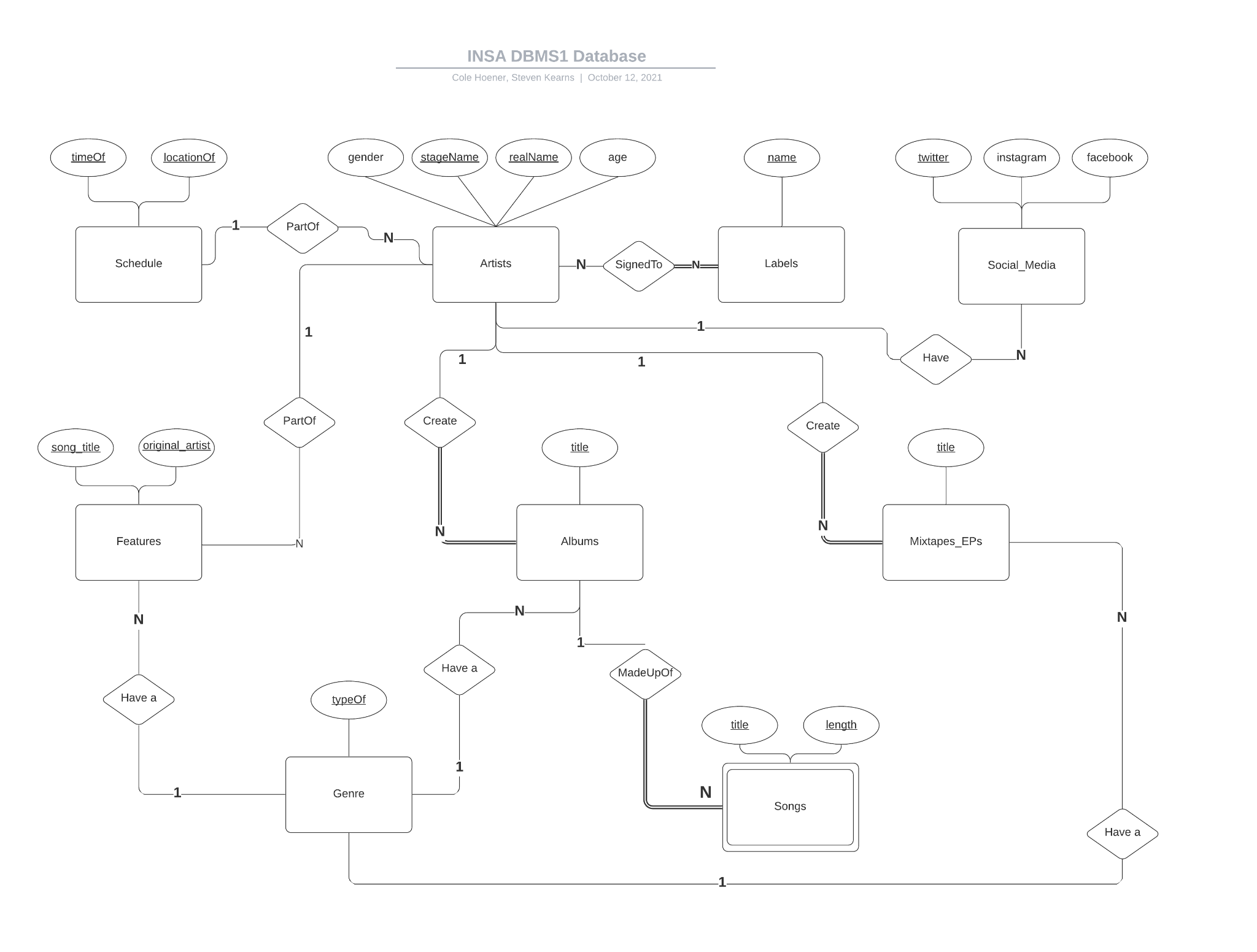
Acquiring and Generating Data

We will be acquiring the data to populate our schema from the official Coachella flyer and from an artists website, MusicBrainz. We will be using the line up from the Saturday of the festival in 2019 to populate the schema for our database. There are various attributes available to use for each artist; however, we will be focusing on the name of the artist, the artists albums, songs, forms of social media, and their label record. We will be manually populating the database from the official Coachella poster to figure out what artists play on Saturday and MusicBrainz website to obtain information about the artist such as songs and forms of social media. Using this information, we will be populating an Excel spreadsheet to later download as a CSV file and populate the SQL database by parsing through it.

Business Rules to Hold

In creating the data for our application, there are a few rules that need to be followed. Each artist has a real name, a stage name, age, gender, a label, and no two artists can have the same stage name. Artists are a part of a label. A label is described by a unique name. Artists create Albums and Mixtapes/EP’s and they are a part of a Feature. Artists will be a part of a schedule to keep track of the time and location that the artist needs to perform at. A schedule has a location and a time, and no two artists can perform at the same time and location. Albums have a unique album title and belong to a genre. Albums are made up of songs. Mixtapes/EP’s have a unique title and belong to a genre. No two mixtapes/EP’s have the same name. Features have a song title and the name of the original artist. Features belong to a genre and no two songs titles and original artist are the same. Songs have a title and the length of the song in seconds. No two songs have the same title and length in seconds. Artists have various social media platforms. The platform will have twitter, instagram, and a unique facebook.

Conceptual Design



The ER diagram holds up to the business rules based on direct understanding of the rules and logical reasoning implemented along the way. Key words within the business rules such as ‘only one’ or ‘unique’ when talking about relationships translates over to a 1-1 or 1-N relationship in the ER diagram. Any attributes listed in the business rules are directly applied to each entity. Primary keys were determined through the understanding of what each attribute meant and how unique they would be. Some primary keys were strictly outlined by the business rules saying which attributes were unique or not. Any cascading deletions were based on logical database design for we did not want data that is not tied to anything filling the database.

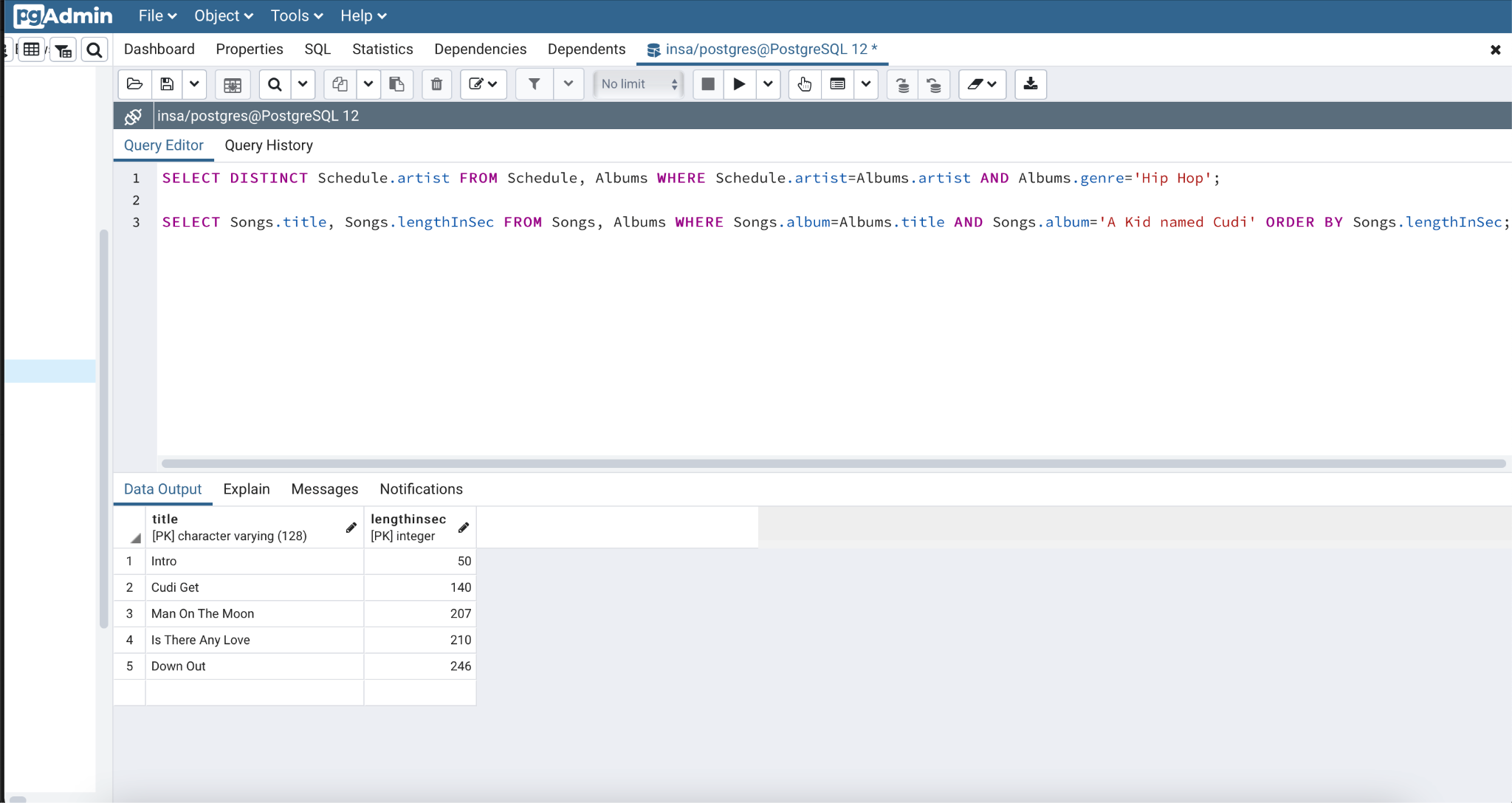
Logical Design

The logical design can be found in the script called CreateDB.sql

The script works in this order:

1. DROP existing tables with the table names we are using
2. CREATE each table
3. INSERT all the values from the data we gathered
4. UPDATE a tuple from the social media table
5. DELETE a tuple from the social media table
6. INSERT that same tuple we deleted in the previous step

While converting the conceptual design to the logical design, there were a few things that needed to be changed but many things were kept the same. The first step taken was just to get every table created statement down to follow the conceptual design. For example, each thought of attribute was within each table and the primary keys were assigned. Overall, each conceptualized primary key and attribute worked within each table. A few of the attribute names needed to be tweaked for they were already keywords in SQL (according to VSCode). After that we simply started creating relations between the tables. The most popular choice to do so was through references. This led to some of the already established tables requiring to add an attribute or two especially in the case two tables had a 1-1 relationship or 1-N. For example, the table ‘Artist’ needed a foreign key constraint on the labels which required a new attribute of “label” to be implemented into ‘Artist’, ensuring the N-N relationship. This process continued through the logical design according to the conceptual design. We did not find any road block in terms of rules that were impossible to hold. Our end product for the logical design consisted of the eternity of the conceptual design with some added attribute to each table/entity. Each choice we made upholds the business rules and ensures that unwanted overlapping or unwanted null values does not happen. We learned from this larger scale experience of converting a conceptual design to a logical design that some constraints and attributes are bound to be added to uphold any business or conceptual rules.



Sample queries

1. Which artists on the schedule have made a Hip Hop album?
   1. SELECT DISTINCT Schedule.artist FROM Schedule, Albums WHERE Schedule.artist=Albums.artist AND Albums.genre='Hip Hop';
2. What songs are in the album A Kid named Cudi, from shortest to longest song length?
   1. SELECT Songs.title, Songs.lengthInSec FROM Songs, Albums WHERE Songs.album=Albums.title AND Songs.album='A Kid named Cudi' ORDER BY Songs.lengthInSec;
3. How many songs are in each album, starting from highest song count to lowest?
   1. SELECT Albums.title, COUNT(Songs.title) FROM Albums, Songs WHERE Albums.title=Songs.album GROUP BY Albums.title ORDER BY COUNT(Songs.title) desc;
4. What are all of the artists' social media, sorted by their Facebook?
   1. SELECT Artist FROM SocialMedia ORDER BY Facebook;
5. For each label, how many labels are associated with Atlantic records?
   1. SELECT name, COUNT (name) FROM Labels WHERE name='Atlantic Records' GROUP BY (name);
6. How many artists are signed to each label, from highest to lowest?
   1. SELECT Labels.name, COUNT (Artists.stageName) FROM Labels, Artists WHERE Labels.name=Artists.label GROUP BY (Labels.name) ORDER BY COUNT(Artists.stageName) desc;
7. Which artists have songs over 300 seconds?
   1. SELECT DISTINCT A.Artist as Name FROM Albums A, Songs S WHERE A.title=S.album AND S.lengthInSec>=300;
8. On average, how long are songs in each genre?
   1. SELECT G.typeOf as Genre, AVG(S.lengthInSec) FROM Genre G, Songs S, Albums A WHERE G.typeOf=A.genre AND A.title=S.album group by G.typeOf;
9. How many albums are under each genre?
   1. SELECT G.typeOf as Genre, COUNT(A.title) FROM Genre G, Albums A WHERE G.typeOf=A.genre group by G.typeOf;