# Data Model and Datasets Considerations

## Resources

* [INTERNAL\_Data Dictionary\_08.11.21: INTERNAL Data Dictionary 08.11.21](https://quip-amazon.com/9qYMAILHNnEi#TCB9CAA63Aq)
  + Contains the trends and information about how the data is presented
* [EXTERNAL\_Data Dictionary\_08.11.21: EXTERNAL\_Data Dictionary\_08.11](https://quip-amazon.com/KD2lAPntoNG2#Pba9CACJ6sO)
  + Explanations about how the data works and the roles used in QuickSight Q

## Data Sets

* [SAMPLE\_Streams\_2018\_2019](https://quip-amazon.com/3IQtA3eEdKab) (the first 200 lines of the Streams Table)
* [Users\_2018\_2019](https://quip-amazon.com/3obtAacRLMjI)
* [Subscribers\_2018\_2019](https://quip-amazon.com/6rZZAJfbzgnZ)
* [Songs\_2018\_2019](https://quip-amazon.com/2J3lAp2462YU)
* [Artists\_2018\_2019](https://quip-amazon.com/xQUkAkrpb0G2)

## Overview

This model is intended to be intuitive for QuickSight Q users by performing basic functions for a streaming service. To display the unique selling points of QuickSight Q, the model has many opportunities for measures and cross-table calculations. However, to keep the model simple for usage and maintenance, there are the following constraints:

* The model is a normalized relational database due to the widespread usage of SQL and my personal experience with developing this type of database. With multiple tables, it shows users how QuickSight Q expedites the process of joining tables and links semantically related data.
* Although foreign keys and primary keys are used in the datasets, marked in the diagram with underline for primary key and strikethrough for ~~foreign key~~. The primary keys in the data model are the main identifiers for each table (exp: song\_id for the Songs table, user\_id for the Users table). The foreign keys are used to refer to the information used in other tables (exp: the Songs table has the foreign key artist\_id to refer to the Artists table, which allows links such as song\_id 1 to be linked to artist\_id 67: from the song\_id alone, one could retrieve information about the artist).
* The only names used in the data model are those approved in the [AWS Style Guide.](https://alpha-docs-aws.amazon.com/awsstyleguide/latest/styleguide/safenames.html) For a full list of the Data Values used, refer to this document.  [Fictitious Names Used](https://quip-amazon.com/xRJ6AIW0qdis)
* Aside from the fictitious names used, all of the data has been created using [Mockaroo](https://www.mockaroo.com/), and automatic random data generator.
  + Despite reaching out to several Amazon teams about acquiring publicly available information for the dataset, I have been unable to as of yet. To move forward with the walkthrough, I created all of the data with the generator.

## Questions

* Which songs were the most popular in 2019?
* Which platform (mobile, desktop, app) was the most popular in 2019?
* What was the subscription revenue in 2019?
* Show me the monthly traffic on each platform in 2019.
* What is the traffic on mobile by country?
* How many subscribers were recruited in 2019 by month?

## Data Model Usage

### General Usage

### Artists

* static list of artists, best left unchanged
* fields of importance: artist id and name
  + being able to pull up the name instead of the artist id would be a nice-to-have for the future→ further abstracts from the data model behind the scenes and provides a more user-friendly experience

### Songs

* static list of songs, can have data added
* field of importance: song\_id and artist\_id
  + song\_id - identifies the song; no song name could be added due to potential licensing issues and to avoid any potential legal problems with manufactured names or translation
  + artist\_id - identifies which artist created the song
    - being able to identify which artist created the song would be an extremely interesting and useful link for Q to be able to make (have not had success in doing so, will continue attempting today)
* potential additions:
  + song release trends: x number of songs released over a period of months to be tracked
  + artist activity (songs release would only be part of this): tracking artist access, listeners, and song releases.

### Users

* increasing list of users, has spikes in Jan, Feb, Jun, Oct for increases (to account for a potential Campaign table)
* fields of importance:
  + user\_id - identifies a unique user

### Subscribers

* increasing list of users, has spikes in Jan, Feb, Jun, Oct (to account for a potential Campaign table)
* fields of importance:
  + subscriber\_id - identifies a unique subscriber
  + user\_id - identifies which user account the subscriber account is linked to (1:1)

### Streams

* rapidly increasing record of users listening to songs, has spikes in Feb, June, Oct
* fields of imporance:
  + stream date and time - not unique, but groups when users listen to music by date.
  + user\_id - information about the user that is listening
  + song\_id - information about the song that is being listened to

### Potential Additions (for the future)

**Campaign Table**

* list of recruitment ads/campaigns, spikes in the users and subscribers list are already in the data to account for this possible addition
* left out for the most recent iteration of the data model for simplicity and to focus more on the streaming traffic of songs
* can also be one type of ad (a potential hierarchy would be ad → campaign or sponsored ad; inheritance in databases)

## Analysis

### Platform Traffic Analysis - Streams

* compares methods of access: mobile, desktop, and app (could also be reduced to mobile and desktop)
  + in this case, the intention is to have website access for mobile and desktop as well as a separate application that could be downloaded (similar to Spotify or Pandora, which have apps as well as a website-usable platform)
* applicable to businesses that have multiple delivery/access methods
  + particularly businesses with websites→ knowing which platform to optimize for is extremely important and often drives UI design decisions (mobile-first vs desktop-first)
* simplification: the data might come from 3 different sources instead of merely being tagged differently; they were just given the values mobile, desktop, and app to reduce the number of tables
* Growth: adding in attributes such as:
  + time spent on the platform (add song duration to the songs table, add the times together)
  + point of access/referrals (where did stakeholders come from? social media, search engine, etc.)

### User Traffic Analysis - Streams

* multiple uses: recruitment, by country/region, by music streaming

### Song Traffic Analysis/Song Popularity- Streams

* observes the number of stakeholders that listen to particular songs

### Artist Popularity

* number of “Monthly Listeners” - placeholder for the number of times people listened to the artist in a given month
  + covers for Q not being able to pull information across tables
  + covers for the ~25k lines used for streams (not enough to be a realistic number).

## Calculations/Calculated Fields

### Subscription Revenue

The Subscription Revenue/Income is the amount earned from purchasing a membership. The subscription revenue will be calculated monthly and yearly with calculated fields.

Previously, it was meant to be a monthly recurring cost for users for realism, but due to scope and inability to develop an expression that would calculate this, the current subscription revenue is annual.

### Song Streaming Costs [out of scope]

Song Streaming Costs are the amount given to artist per stream (royalties) and the amount the artists make per month (monthly payment). This calculated field will be calculated monthly  (anticipated) and yearly.

## Developing Trends in the Data (Preparing Data for Semi-Realistic Visualizations)

I used [Mockaroo](https://www.mockaroo.com/)  to create randomly generated datasets. For trends, I used:

* normal distribution
* random number generation from a range of numbers and decimals
* random date generation from a range of dates.

For the trends shown in the example topic (and in the subsequent visualizations), I randomly generated dates that would be created and used the row numbers as IDs to keep track of identifiers so there would not be any dangling references (Exp: Song ID 10 referring to a nonexistent Artist ID 1000000).

The exact numbers used for the trends are listed in the [INTERNAL\_Data Dictionary\_08.11.21: INTERNAL Data Dictionary 08.11.21](https://quip-amazon.com/9qYMAILHNnEi#TCB9CAA63Aq).

## !Dataset Details and Adding More Data

Note: primary key = main identifier for all of the information in the table; ~~foreign key~~ = identifier for linking information from another table.

The Data Model has 5 tables that each refer to a unique entity (Users, Subscribers, Artists, Songs, Streams). For more information on the exact implementation details, refer to the data dictionary. [Data Dictionary](https://quip-amazon.com/9qYMAILHNnEi)

## Read this before editing the data

In relational databases (which is what this data model is based off of), there are two major errors you can have when editing the data: insertion errors and deletion errors. In general, the errors occur when you are dealing with something of “uniqueness”, unique being something that

Insertion errors can occur when:

* you create something that already exists but should be unique (duplicates)
  + i.e. user\_id 1 is named A, and you want to insert user B with user\_id 1

## Potential Add-Ons (these would need to be a separate entity or calculated field)

* Advertisements
  + can be used for inheritance (sponsored ads, company ads, recruitment, etc.)
* Streaming Costs (currently has fields, but they’re not being used in calculations)
* Annual Profit (calculated streaming costs and subscription + ad revenue)
* Song Playlists and Albums
* User Likes/Dislikes
* User Activity
  + likes/dislikes
  + time spent listening/on the application
  + frequency of visits
  + commonly accessed genres/songs/artists

## Dataset Setup (in case anything is deleted)

Note: if recreating the datasets, the  [INTERNAL Data Dictionary 08.11.21](https://quip-amazon.com/9qYMAILHNnEi) will have additional implementation details.

To set up the tables, I inserted the following rows randomly for the year of 2018:

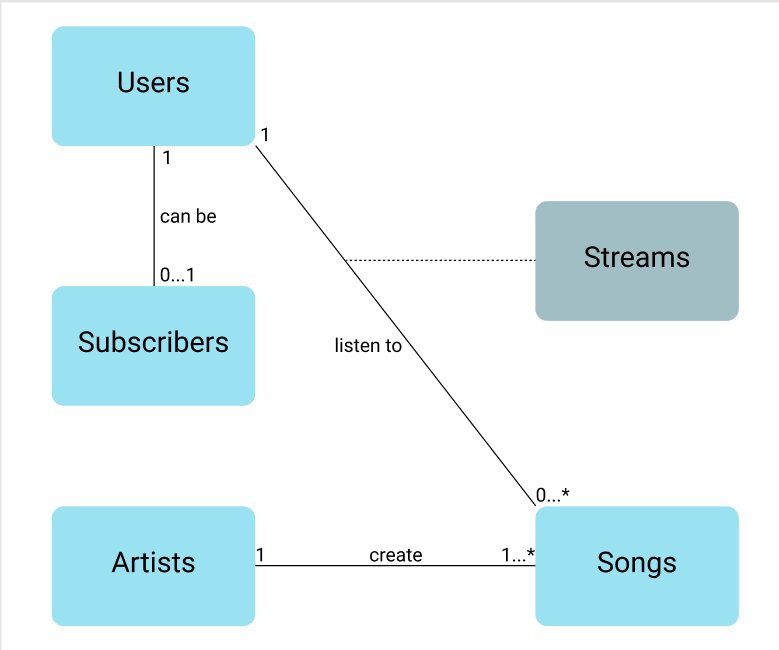
* Users: 2000
* Subscribers: 0
* Artists: 70 (stays the same due to the number of fictional names I was able to use)
* Songs: 2000 (stays the same for simplicity and to focus on the usage of traffic analyses)
* Streams: 6000 (would have liked the overall number to be increased by a factor of ten, but for convenience and space savings, it was limited to this number initially.

The tables should have the following number of rows (in total):

* Users: 4600
* Subscribers: 2300
* Artists: 70
* Songs: 2000
* Streams: 25000
* [SAMPLE\_Streams\_2018\_2019](https://quip-amazon.com/3IQtA3eEdKab) (the first 200 lines of the Streams Table)
  + Streams (date\_time, platform, ~~song\_id, user\_id~~)
  + **Note:** this table does not have a primary key (this is an associative entity), as it refers to the relationship between users and songs (1 user can listen to 1 or more songs).
  + Because this has no identifier, this table is a prime example for showing how Q can link data (for a relational database) and its usefulness for named entities (in the topic, I created a Streams entity so it would recognize the date/time information as a unique stream).
  + This table is meant to be a static simulation of traffic for the music streaming service’s access points (platform of either mobile, desktop, or app), the traffic for songs (song\_id information), and the traffic for users.
  + INSERTING DATA: to reduce insertion errors, be sure that the user\_id and song\_id that is added for each entry exists before the date\_time of the stream (exp: for a date\_time stream on 01.31.20, make sure that the song was released before that day and that the user joined before that day).
    - Example new row:
    - date\_time = 12/01/19
    - platform = app (doesn’t matter here)
    - song\_id = 10 (doesn’t matter here)
    - user\_id = 4500 (4500 users were created by 11/31/19, as per the data dictionary [Data Dictionary 08.11.21](https://quip-amazon.com/9qYMAILHNnEi))
  + TRENDS: to create visualizations of interest, I added a certain number of streams by a factor of 1000 for each month of 2019. In Mockaroo, this can be easily created by changing the date field to a random date within the month.‘
    - Exp: Jan. 2019 has 1000 streams, Feb. 2019 has 2000 streams, etc.
  + FORMATTING: for Q, there are certain date formats used. For the table, I used MM/dd/yyyy.
* [Users\_2018\_2019](https://quip-amazon.com/3obtAacRLMjI)
  + Users (user\_id, country, join\_date, age)
  + FORMATTING: for Q, there are certain date formats used. For the table, I used MM/dd/yyyy.
* [Subscribers\_2018\_2019](https://quip-amazon.com/6rZZAJfbzgnZ)
  + INSERTING DATA: to reduce certain errors,
  + TRENDS: to create visualizations of interest, I added a certain number of streams by a factor of 1000 for each month of 2019. In Mockaroo, this can be easily created by changing the date field to a random date within the month.
  + FORMATTING: for Q, there are certain date formats used. For the table, I used MM/dd/yyyy.
* [Songs\_2018\_2019](https://quip-amazon.com/2J3lAp2462YU)
  + static for the data model to prevent confusion.
  + RECOMMENDED: do not add more data to this table unless necessary.
* [Artists\_2018\_2019](https://quip-amazon.com/xQUkAkrpb0G2)
  + static for the data model to prevent confusion and because 70 is the maximum amount of fictitious names I was able to find that would minimal legal issues (copywriting, licensing, etc.) [Data Values](https://quip-amazon.com/xRJ6AIW0qdis)
  + RECOMMENDED: do not add more data to this table unless necessary.

## Images of Data Model 08.11.21

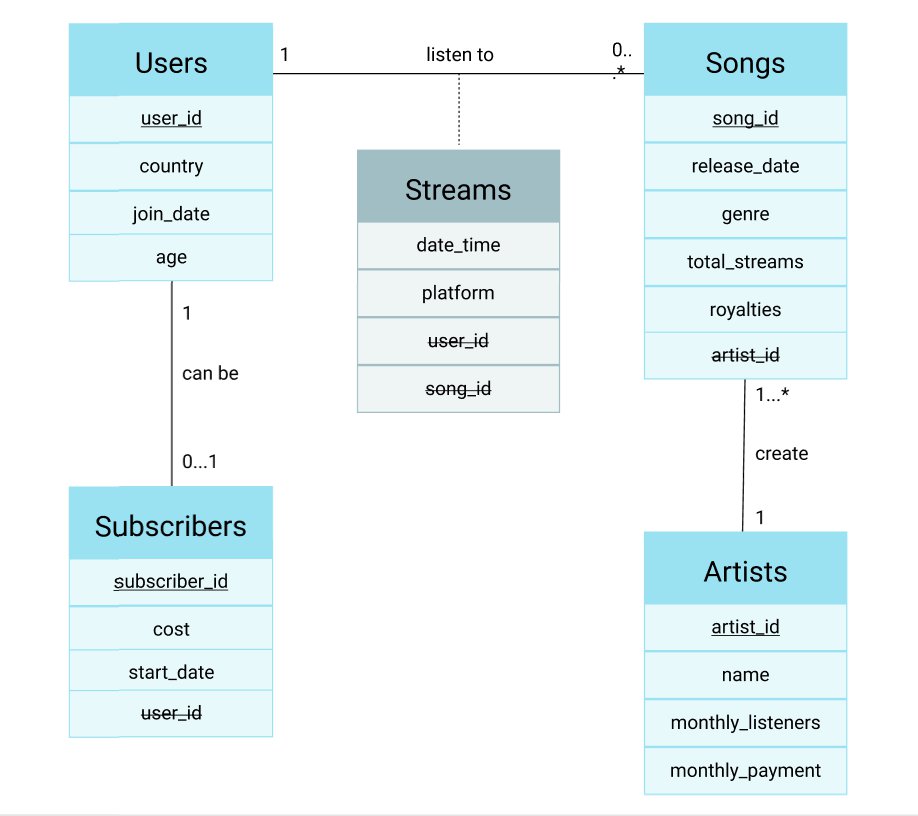
## !Basic Conceptual Model and Relations



### Conceptual Relations:

* 1 User can become a Subscriber (or not)
* 1 User can listen to many songs (or not at all)
* 1 Artist can create 1 to many Songs
* When a User listens to Songs, they Stream music. (the Streams table is an associative entity, which is a bit different than the other entities— User, Songs, etc.— it takes information from the User-to-Song relationship to provide additional data about each individual interaction.

## Logical Model (the data model at a glance)



## Licensing + Usage

Public Dataset Licenses Approve for Internal Use <https://grc.amazon.com/docs/83021>

ML Public Dataset Policy <https://grc.amazon.com/docs/83291#s4>.