Project Outline, Database Outline, ERD, and Schema

DJ's MP3's

a. Overview

We operate an online MP3 store that allows customers to purchase music by the song or by the album. We sell 2,000,000 tracks per year for an average of 5,500 tracks per day. A database driven website will allow us to keep track of unique ***customers*** and record their ***orders*** (a list of downloads) and which ***download_items*** were on their orders. ***customers*** will have the ability to purchase individual ***tracks*** for .99 each or purchase an entire ***album*** for 9.99. ***customers*** will be able to select multiple tracks for purchase or multiple albums, they may not purchase more than one qty of each unique item.

b. Database Outline

customers: records the details of customers we do business with

- customer_id: int, auto_increment, unique, not NULL, Primary Key
- customer_first_name: varchar, not NULL
- customer_last_name: varchar, not NULL
- customer_phone: char(11), not NULL
- customer_street: varchar, not NULL
- customer_apt: varchar
- customer_city: varchar, not NULL
- customer_state: char(2), not NULL
- customer_zip: char(5), not NULL
- customer_email: varchar, not NULL
- credit_card_number: char(16), not NULL
- credit_card_exp: char(4), not NULL
- Relationship: A 1:M relationship between customers and orders is implemented with customer_id as a Foreign Key inside of orders.

orders: records the details of orders placed by customers.

- order_number: int, auto_increment, unique, not NULL, Primary Key
- customer_id: int, not NULL, Foreign Key
- order_date: date, not NULL
- credit_card_number: char(16), not NULL
- credit_card_exp: char(4), not NULL
- total_cost: money, not NULL
- Relationship: A M:M relationship between orders and tracks is implemented with the creation of a download_item composite entity that splits the M:M relationship into two 1:M relationships (one track to download can belong to many orders and one order can contain many tracks) .

download_items: Is one track that can belong to either orders of full albums or orders for individual tracks. When an album is selected for download, then all individual tracks with that album as attribute are downloaded by querying the DB for all tracks with the unique album_id.

- order_number: int, not NULL, part of Concatenated Primary Key, Foreign Key
- track_id: int, not NULL, part of Concatenated Primary Key, Foreign Key
- single: bool, not NULL (indicates if item being downloaded is individually downloaded track or track downloaded as part of full album, if True, then higher price is added to subtotal. If False, then track is part of album download and lower price is used for subtotal)
- Relationship: A M:M relationship between tracks and orders is handled by creating a download_item composite entity that allows only one song to be on one download_item and many unique download_items to appear on one unique order. Takes order_number and track_id as Foreign Keys to handle M:M relationship between orders and tracks.

tracks: individual tracks to be ordered, tracks have one and only one artist as do albums. (album_id is now NULLable to reflect fact that a track can be a single release not belonging to an album)

- track_id: int, auto_increment, unique, not NULL, Primary Key
- title: varchar, not NULL
- artist_id: int, not NULL, Foreign Key
- album_id: int, Foreign Key
- ind_price: int, precision (hundredths place), not NULL
- album_price : int, precision (hundredths), not NULL (album price is lower than price of individual download)
- Relationship: 1:M relationship between artist and track where one artist produces many tracks. Each track corresponds to one and only one artist. artist_id is a Foreign Key linking tracks to artists. A 1:M relationship exists between albums and tracks, a track may have 0 or 1 album and an album may have 1 or many tracks.

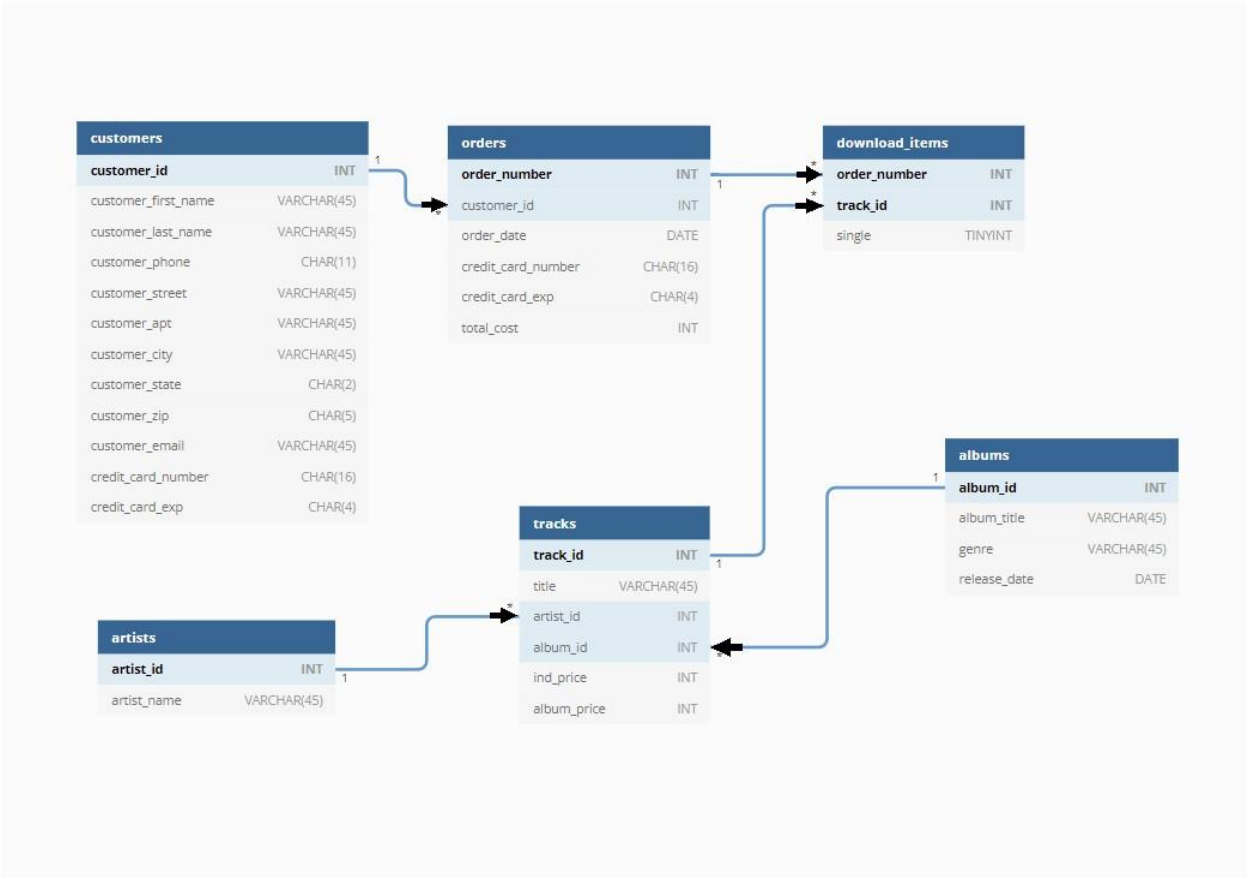
artists: Identifies unique artists and allows tracks to be queried by artists.

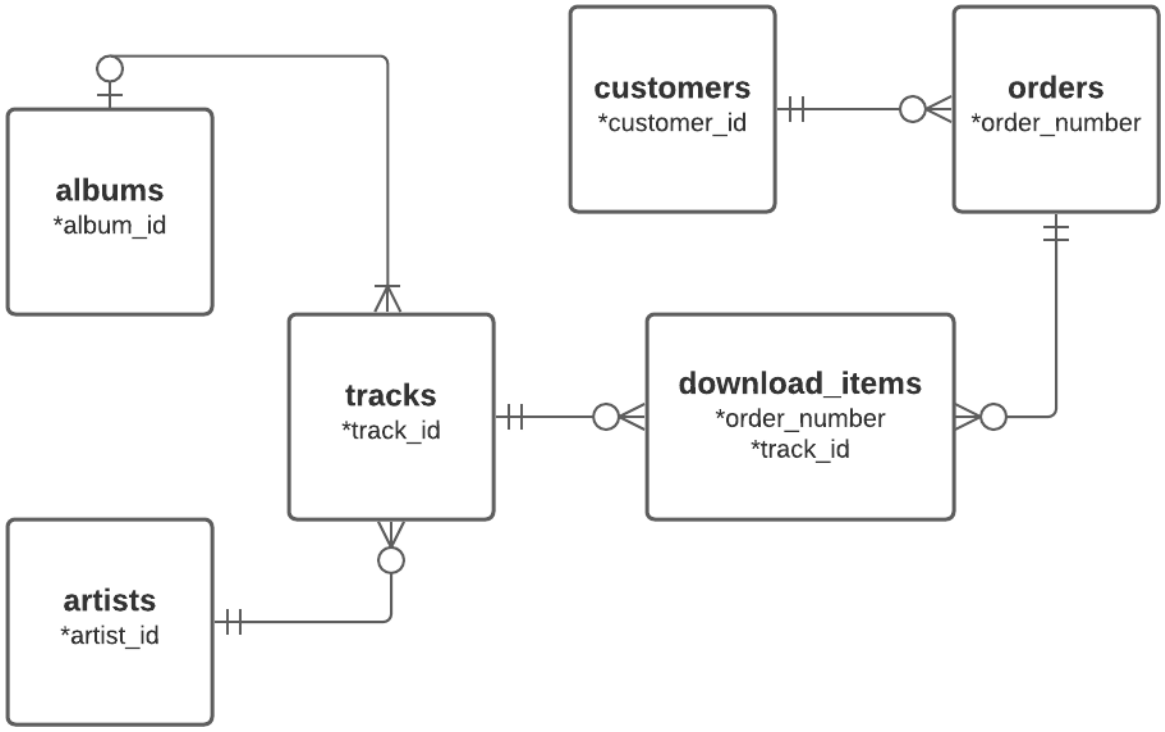
- arist_id: int, auto_increment, unique, not NULL, Primary Key
- artist_name: varchar, not NULL
- Relationship: A 1:M relationship exists between artists and tracks. artist_id is a Foreign Key inside tracks.

albums: Identifies unique albums and relates albums to artists, allows tracks to be queried by album_id for when purchasing a whole album.
(* updated so that artist_id is no longer an attribute)

- album_id: int, auto_increment, unique, not NULL, Primary Key
- album_title: varchar, not NULL
- genre: varchar
- release_date: date
- Relationship: A 1:M relationship exists between albums and tracks. There is no longer a direct relationship between albums and artists as albums and artists are joined by the artist_id FK attribute under ‘tracks’. A 1:M relationship exists

between albums and tracks, a track may have 0 or 1 album and an album may have 1 or many tracks.

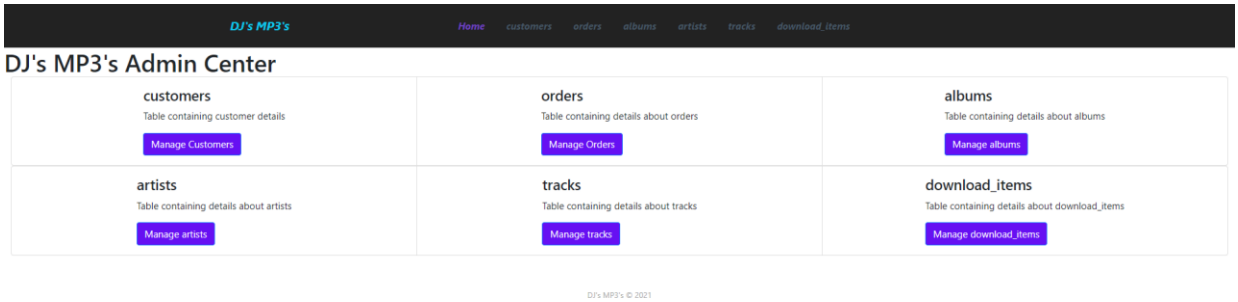




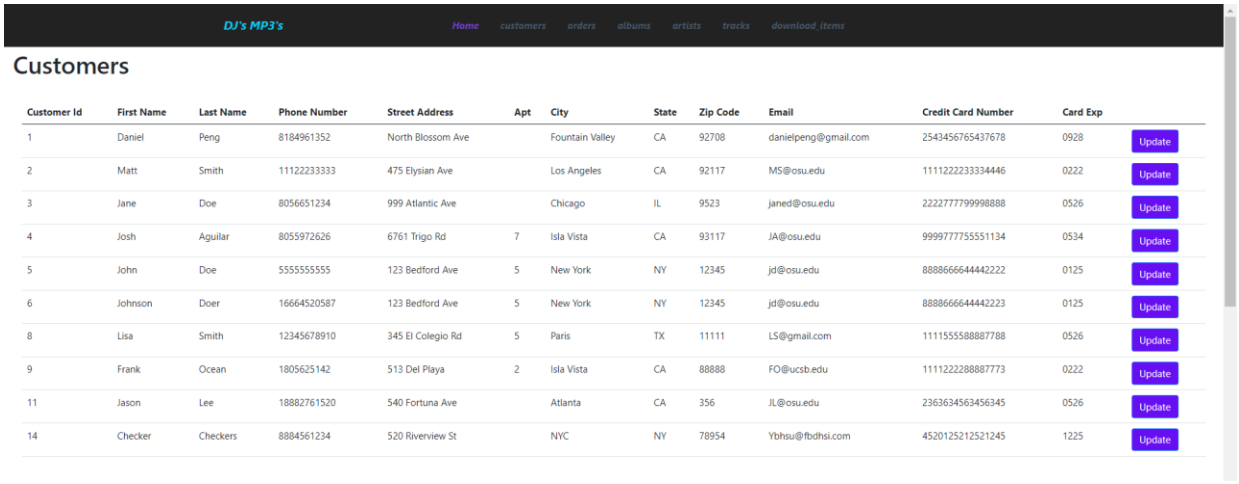
*UI screen captures below

UI Screen Captures

Home page: No CRUD functionality. Allows for convenient navigation across other pages



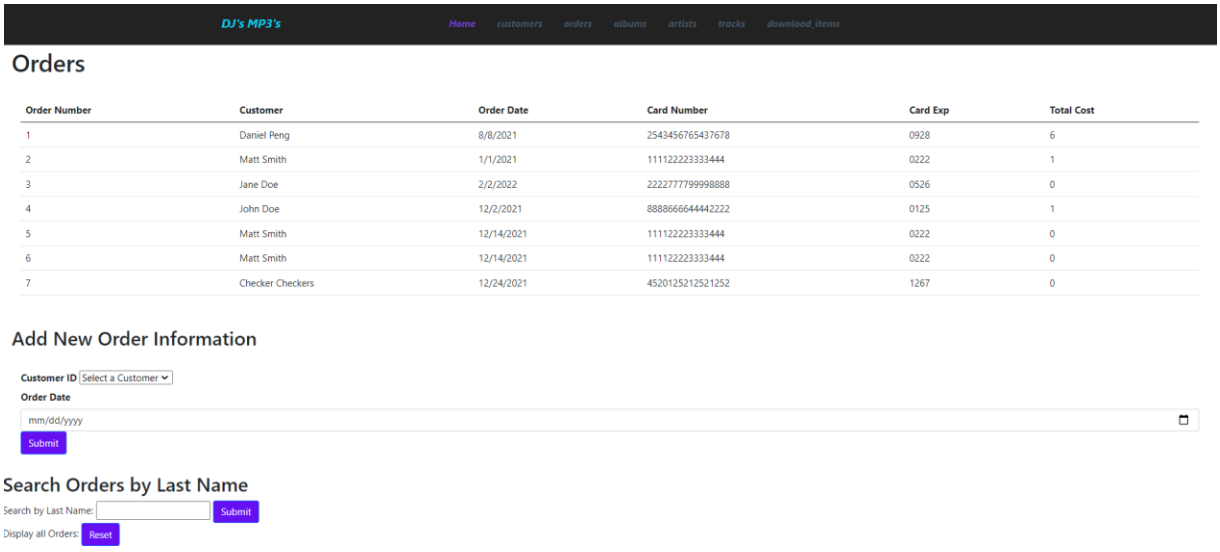
READ/BROWSE/DISPLAY Customers page; also UPDATE/EDIT Customers page (top half of page)



Same page as above (bottom half); CREATE/INSERT/ADD NEW customers page

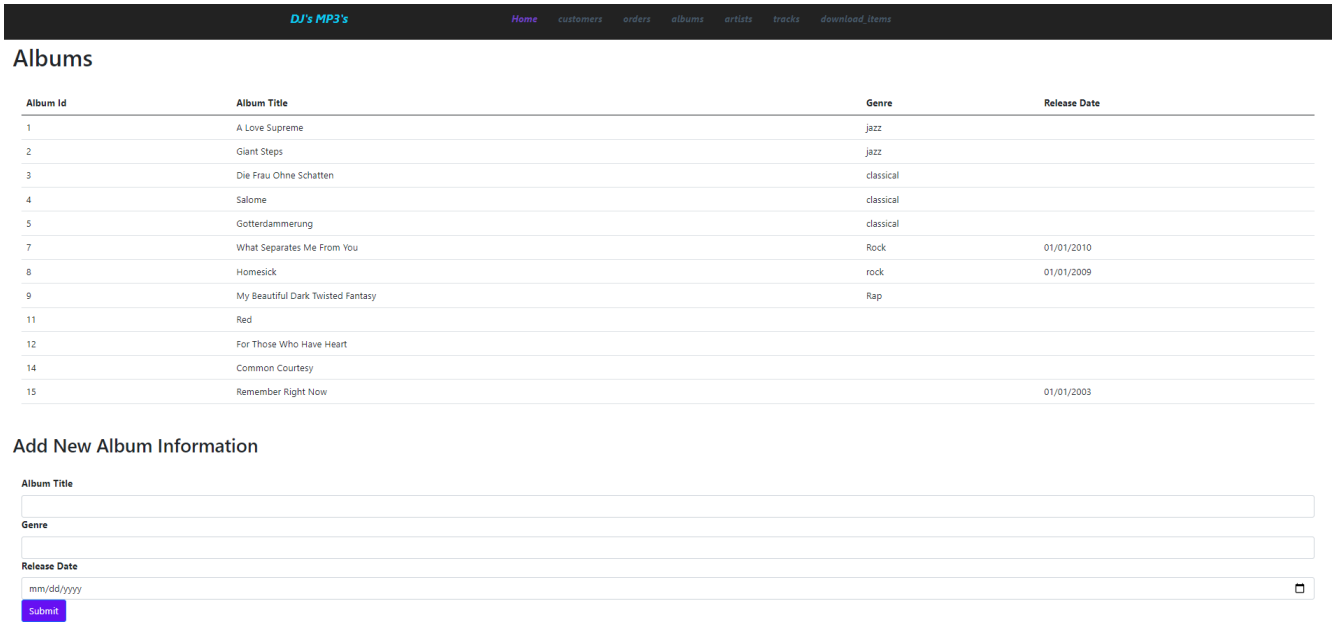
READ/BROWSE/DISPLAY orders page (top); CREATE/INSERT/ADD NEW orders page (middle)

SEARCH/BROWSE orders page (bottom)



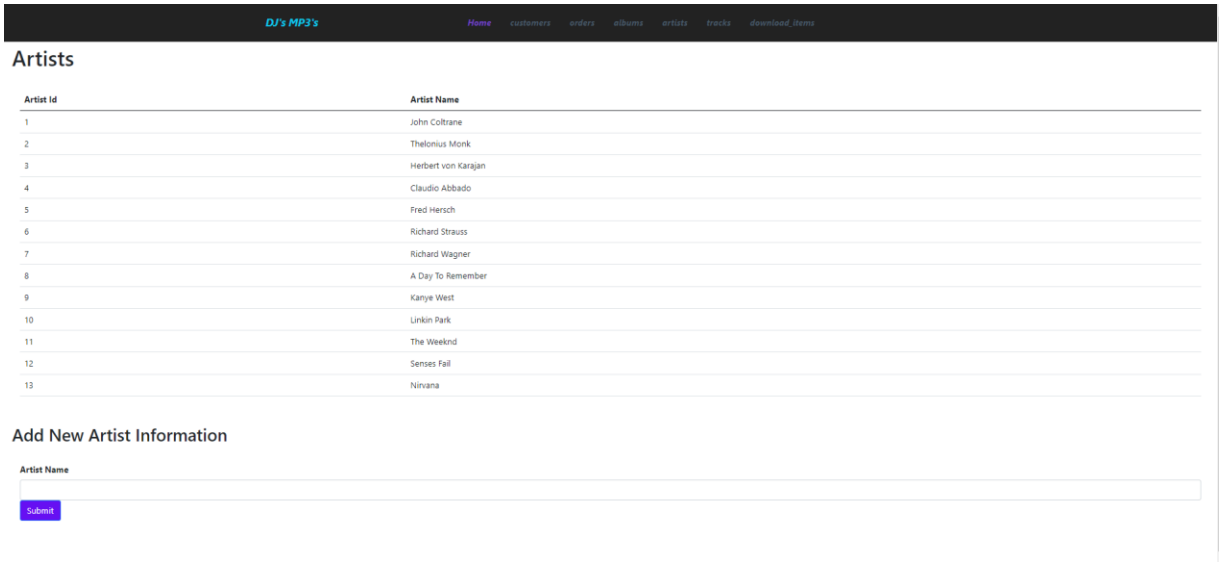
READ/BROWSE/DISPLAY albums page (top);

CREATE/INSERT/ADD NEW albums (bottom)



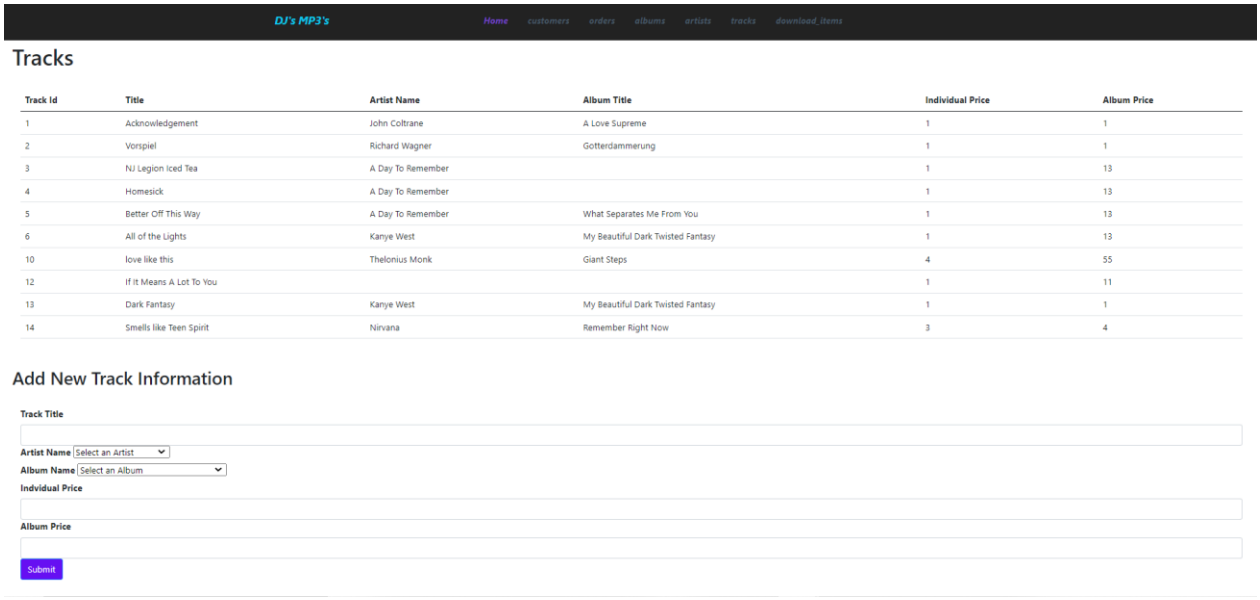
READ/BROWSE/DISPLAY artists page (top);

CREATE/INSERT/ADD NEW artists page (bottom)



READ/BROWSE/DISPLAY tracks page (top);

CREATE/INSERT/ADD NEW tracks page (bottom)



READ/BROWSE/DISPLAY/DELETE tracks page (top);

CREATE/INSERT/ADD NEW tracks page (bottom)