Names
- Jakob Peters
- Brayden Aldrich

Site link: http://flip2.engr.oregonstate.edu:3831/

Executive summary of changes:

In the initial phase of our project we made changes to our schema, SQL, and project overview. In the schema we created two new tables: Purchase_Items and Items. These would be the link between what we sell and Purchases. Items would house each Album, Song, and Device, and Purchase_Items would be a table of each Item that is purchased. Additionally, we created a table called Labels, which would satisfy the requirement of partial participation with Artists – as some Artists are independent. In terms of SQL, we implemented an ON DELETE CASCADE for every foreign key in our database. As for our overview, we went into more detail about the numbers involved with the operation.

In the next step, we implemented our front-end using standard HTML and a few javascript functions. We also made changes to our schema surrounding Albums and Songs. The first specifying that in this universe, singles would have a dummy Album associated with it. For the second change, we received feedback on the case where there can be multiple Artists per Abum. In Albums we added the specification that Albums would assume a primary artist or only one artist.

After this step, we changed a lot about our site. We moved from HTML to handlebars and we also implemented a good portion of the CRUD functionalities purely on the backend using NodeJS, as well as implementing all the functionalities for Purchase_Items. We also made the our code more modular, so it can be used by many different tables. We also added a bit of css so the site would be easier on the eyes.

For the next step, we updated our DML file to include the queries we were using in our backend. We also made dropdown menus use data from foreign tables so that it would be easier to use. During this step, we fixed a good deal of bugs, so almost all features in the site were usable.

Lastly, we changed how the dropdown menus work in Purchase_Items. Instead of selecting a purchase_id and an item_id, users now select the pairs from a single dropdown menu, eliminating the chance to select an entity that doesn't exist. This fixed on bug that appeared from a faulty SQL query. We also fixed the remaining bugs. At this stage, our site is fully functional.

Title: The Music Shop

Overview:

The Music Shop is a record shop in Toronto hosting a catalog of around 500,000 records, depending on current trends. The shop also stores about 2,000 thousand record

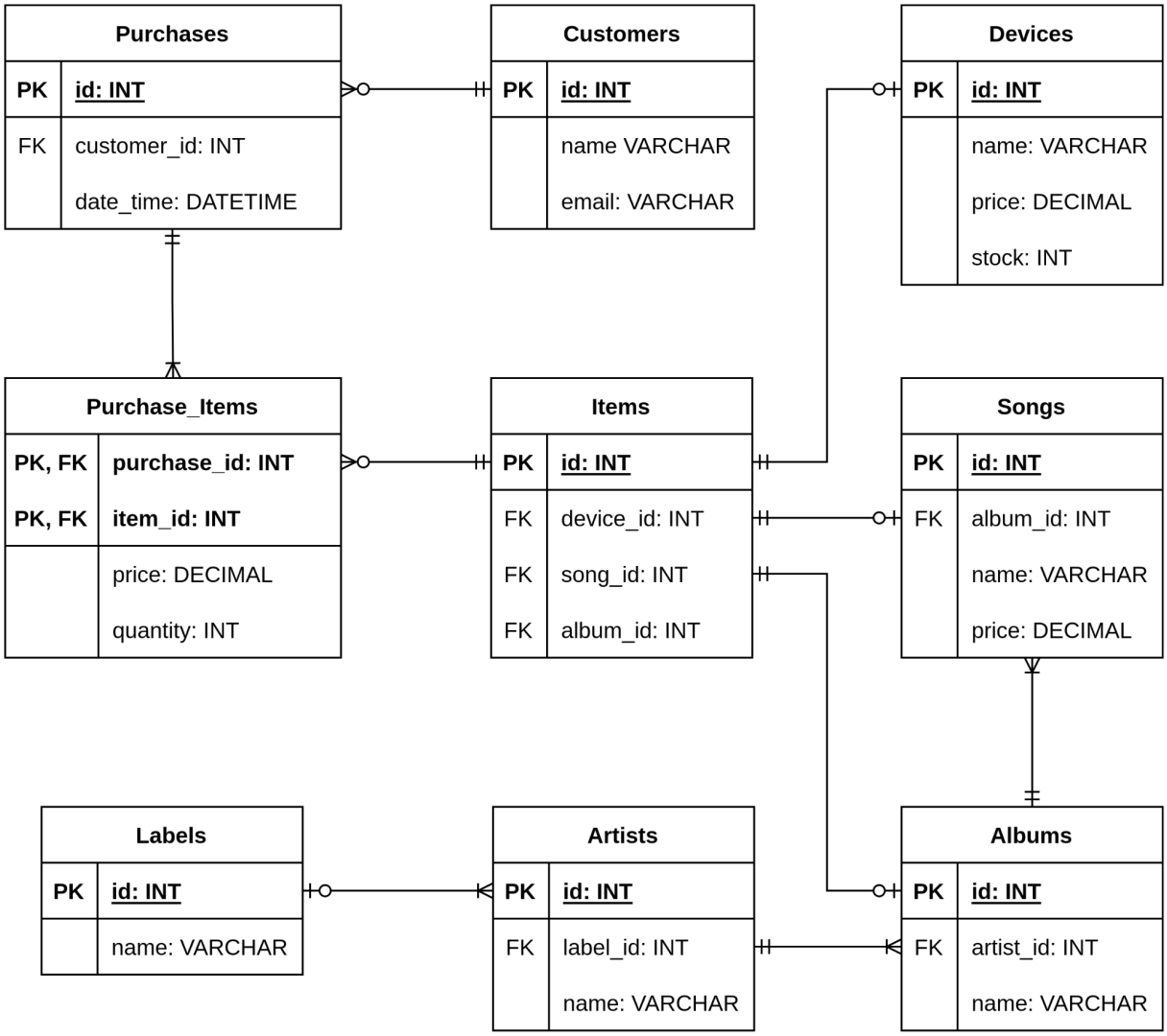
players, 10,000 blank records, and 5,000 usb drives at a given time. The shop brings in around $800,000 annually. Customers can sign up for our mailing list without making a Purchase. Customers can buy music or Devices. They can buy Songs and save them on a USB flash drive, or else Albums saved to a blank record. Besides music, Customers also have the ability to purchase record players. Customers can also cancel orders. We need to keep track of inventory Items (Devices, Albums, and Songs) and sales (Purchases). Customers may make Purchases, Purchases also have at least one Item, each Item is either a Device, Song, or Album, Songs have an associated Album (even if it is a dummy album for singles), each Album has an Artist (we assume either a single or primary artist), and Artists may or may not have an associated Label. We may also have a Song, Album, or Device that is in stock, but is not an Item for sale. Once we have a Song or Album, we will always have it available. We need a relational database to efficiently store and manipulate this data. The website will interface with that database, allowing the store managers to select from and insert data into each entity, cancel orders (remove rows from Purchase_Items), and change/remove the affiliation of Artists with Labels.

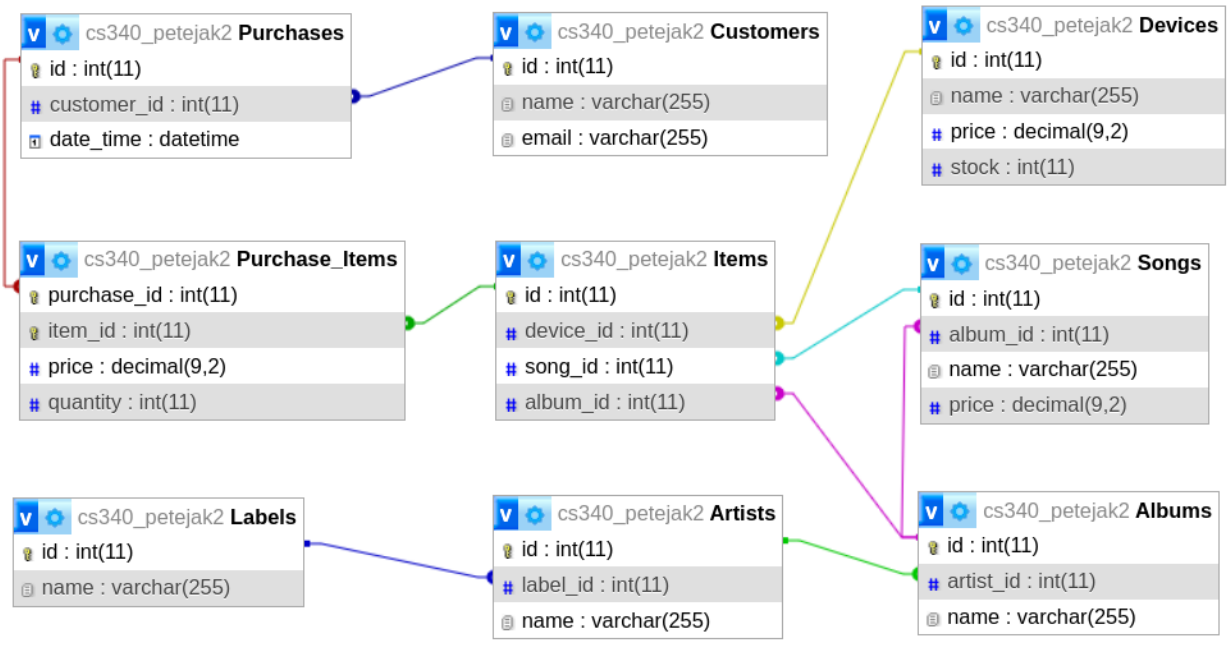
Database Outline:

- Customers
    - People who make a purchase and/or sign up for our mailing list
    - Relationships
        - 1:0|M with Purchases
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - name: VARCHAR, can be NULL
        - email: VARCHAR, can be NULL
- Purchases
    - A customer purchasing at least one product
    - Relationships
        - 0|M:1 with Customers
        - 1:1|M with Purchase_Items
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - customer_id: INT, not NULL
        - date_time: DATETIME, not NULL
- Purchase_Items
    - The intersection table for Purchases and Items, each row is an instance of an item being purchased, with the price at the time of purchase
    - Semantic constraint: The price must be greater than or equal to zero
    - Relationships
        - 1|M:1 with Purchases
        - 0|M:1 with Items
    - Attributes
        - purchase_id: INT, not NULL, PK, FK
        - item_item: INT, not NULL, PK, FK
        - price: DECIMAL, not NULL

        - quantity: INT, not NULL
- Items
    - The items in stock that are for sale
    - Semantic constraint: only one FK is not NULL
    - Relationships
        - 1:0|M with Purchase_Items
        - 1:0|1 with Songs
        - 1:0|1 with Albums
        - 1:0|1 with Devices
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - song_id: INT, can be NULL, FK
        - album_id: INT, can be NULL, FK
        - device_id: INT, can be NULL, FK
- Devices
    - The devices in stock; currently record players, blank records, and USB flash drives
    - Semantic constraint: price and stock must both be greater than or equal to zero
    - Relationships
        - 0|1:1 with Items
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - name: VARCHAR, not NULL
        - price: DECIMAL, not NULL
        - stock: INT, not NULL
- Songs
    - The songs in stock
    - Semantic constraint: price must be greater than or equal to zero
    - Relationships
        - 0|1:1 with Items
        - 1|M:1 with Albums
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - album_id: INT, not NULL, FK
        - name: VARCHAR, not NULL
        - price: DECIMAL, not NULL
- Albums
    - The albums in stock
    - Semantic constraint: price must be greater than or equal to zero
    - Relationships
        - 0|1:1 with Items
        - 1:1|M with Songs
        - 1|M:1 with Artists
    - Attributes

        - id: INT, auto increment, not NULL, PK
        - artist_id: INT, not NULL, FK
        - name: VARCHAR
- Artists
    - An artist that has an album in our inventory
    - Relationships
        - 1:1|M with Albums
        - 1|M:0|1 with Labels
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - label_id: INT, can be NULL, FK
        - name: VARCHAR, not NULL
- Labels
    - The labels that have artists in our discography.
    - Relationships
        - 0|1:1|M with Artists
    - Attributes
        - id: INT, auto increment, not NULL, PK
        - name: VARCHAR


Purchases
PK
id: INT
FK
customer_id: INT
date_time: DATETIME
Customers
PK
id: INT
name VARCHAR
email: VARCHAR
Devices
PK
id: INT
name: VARCHAR
price: DECIMAL
stock: INT
Purchase_Items
PK, FK
purchase_id: INT
PK, FK
item_id: INT
price: DECIMAL
quantity: INT
Items
PK
id: INT
FK
device_id: INT
FK
song_id: INT
FK
album_id: INT
Songs
PK
id: INT
FK
album_id: INT
name: VARCHAR
price: DECIMAL
Labels
PK
id: INT
name: VARCHAR
Artists
PK
id: INT
FK
label_id: INT
name: VARCHAR
Albums
PK
id: INT
FK
artist_id: INT
name: VARCHAR



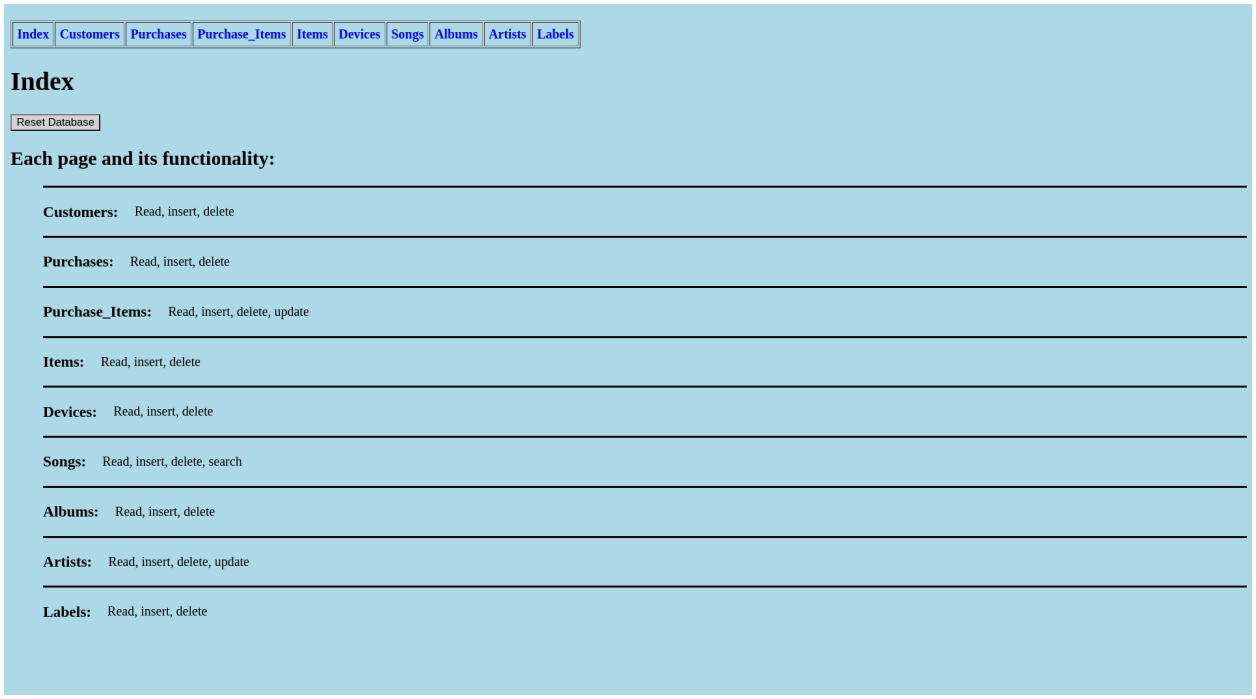
Customers			
id	**name**	**email**	
1	Thomas K.	tomk@gmail.com	
2	Brianna W.	briannaw@gmail.com	
3	Kyle C.	kyle@gmail.com	
4	James A.	james12@gmail.com	
5	NULL	NULL	
Purchases			
id	**customer_id**	**date**	
1	2	2023-01-12 08:04:02	
2	3	2023-01-14 16:08:04	
3	1	2023-01-14 11:57:12	
4	4	2023-01-15 16:32:08	
5	4	2023-01-16 16:37:18	
6	5	2023-01-16 17:17:55	
Labels			

id	**name**		
1	Reprise Records		
2	Vagrant Records		
3	Warner Records		
Artists			
id	**label_id**	**name**	
1	1	Green Day	
2	NULL	Taylor Swift	
3	2	The Get Up Kids	
4	3	Disturbed	
Albums			
id	**artist_id**	**name**	
1	1	American Idiot	
2	2	Midnights	
3	3	Something to Write Home About	
4	4	Immortalized	
Songs			
id	**album_id**	**name**	**price**
1	1	American Idiot	1.00
2	1	Jesus of Suburbia	0.60
3	1	Holiday	1.00
4	2	Anti-Hero	1.00
5	3	Ten Minutes	1.00
6	4	The Sound of Silence	1.20
Devices			

id	name	price	stock
1	record_player	80	2000
2	blank_record	25	10000
3	usb_drive	10	5000
Items			
id	**device_id**	**album_id**	**song_id**
1	1	NULL	NULL
2	2	NULL	NULL
3	3	NULL	NULL
4	NULL	1	NULL
5	NULL	2	NULL
6	NULL	3	NULL
7	NULL	NULL	1
8	NULL	NULL	2
9	NULL	NULL	3
Purchase_Items			
purchase_id	**item_id**	**price**	**quantity**
1	1	240.00	3
1	2	50.00	2
2	3	10.00	1
3	4	2.60	1
4	5	2.00	2
5	6	1.00	1
6	7	1.00	1

UI Overview:

Index page (Reset database functionality)



READ INSERT DELETE Customers page

READ INSERT DELETE Purchases page

Index | Customers | Purchases | Purchase_Items | Items | Devices | Songs | Albums | Artists | Labels

Purchases

Insert

Customer Name Date-Time mm/dd/yyyy, --:-- -- Reset Submit

Delete

ID Reset Submit

Table

id	customer_id	date_time
1	2	Thu Jan 12 2023 08:04:02 GMT-0800 (Pacific Standard Time)
2	3	Sat Jan 14 2023 16:08:04 GMT-0800 (Pacific Standard Time)
3	1	Sat Jan 14 2023 11:57:12 GMT-0800 (Pacific Standard Time)
4	4	Sun Jan 15 2023 16:32:08 GMT-0800 (Pacific Standard Time)
5	4	Mon Jan 16 2023 16:37:18 GMT-0800 (Pacific Standard Time)
6	5	Mon Jan 16 2023 17:17:55 GMT-0800 (Pacific Standard Time)

READ INSERT DELETE UPDATE Purchase_Items page

Index | Customers | Purchases | Purchase_Items | Items | Devices | Songs | Albums | Artists | Labels

Purchase_Items

Insert

Purchase ID Item ID Quantity Reset Submit Query

Delete

Purchase and Item IDs Reset Submit Query

Update

Purchase and Item IDs Quantity Reset Submit Query

Table

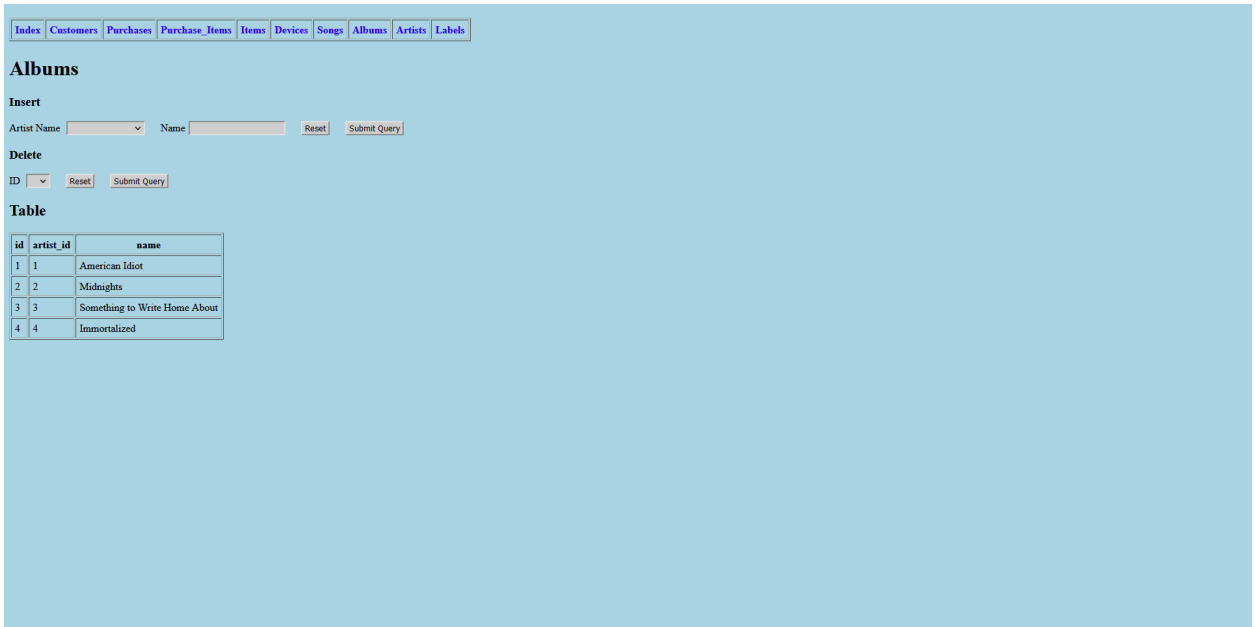
purchase_id	item_id	price	quantity
1	1	240.00	3
1	2	50.00	2
2	3	10.00	1
3	4	2.60	1
4	5	2.00	2
5	4	2.60	1
5	6	1.00	1

READ INSERT DELETE Items page

READ INSERT DELETE Devices page

READ INSERT DELETE SEARCH Songs page

READ INSERT DELETE Albums page



READ INSERT DELETE UPDATE Artists page

READ INSERT DELETE Labels page