University of British Columbia, Vancouver
Department of Computer Science

Project Description

This is a system for theatre management that includes the use of an SQL database. The intended use case for this project is for a theatre chain to manage its locations and its movies on the customer side of the business. This project will allow theater owners to keep track of where the theatre is located, which films, film schedules, the auditoriums where the films are playing, the capacity of the auditoriums, customer subscriptions (including theatre SCENE points (or something analogous)) and theatre guest purchases. Theatre guests and employees will have different access privileges to the database. Guests will be able to buy tickets, buy food, and view movie showings. Using the database employees will be able to create new movie showings, create memberships, view tickets, and view guest information.

Database Specification

Keeping track of the films playing, the film schedules, the auditorium capacities and the customer information, will all require use of a database and a way to manage that database. A feature of this project is the customer will be able to have a subscription which will include benefits in the form of discounts and points. In order to keep track of these points and benefits, a database will be required.

Project Repository and Link to the PHP

PHP Link : https://www.students.cs.ubc.ca/~gsebass1/theatres.php
Repo : https://github.students.cs.ubc.ca/CPSC304-2021W-T1/project_h8w1b_n7v2b_r5i2b.git

Changes to Schema

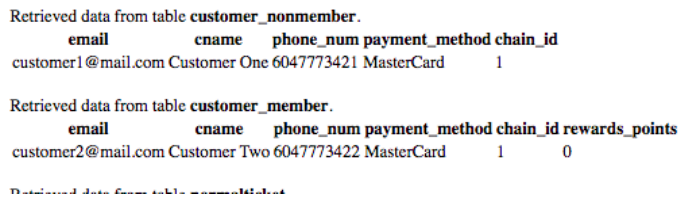
- Removed some of the email dependencies between purchases to facilitate insertion and purchases without prior registration.
- Made seats ('rec_seat' and 'nonrec_seat') contain the time they are being used and consequently, dependent on 'movieshowing' instead of being directly dependent on 'auditorium'.

University of British Columbia, Vancouver
Department of Computer Science

<u>SQL Queries</u>

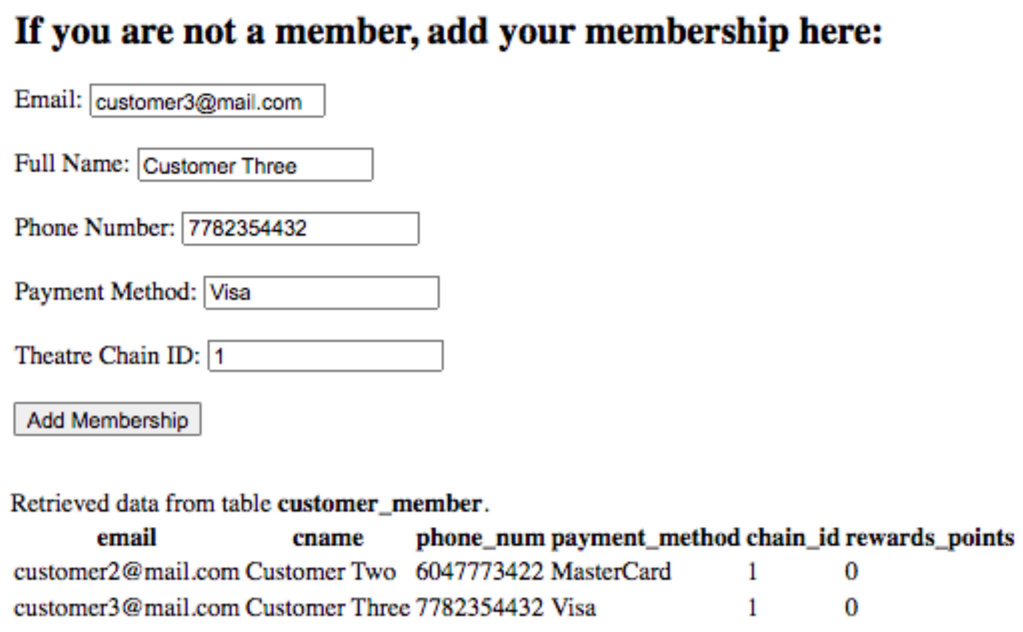
Queries: INSERT Operation

insert into customer_member values('customer3@mail.com', 'Customer Three', '7782354432','Visa',1)



```
Retrieved data from table customer_nonmember.
       email              cname       phone_num payment_method chain_id
customer1@mail.com Customer One 6047773421 MasterCard        1

Retrieved data from table customer_member.
       email              cname       phone_num payment_method chain_id rewards_points
customer2@mail.com Customer Two 6047773422 MasterCard        1          0
```



If you are not a member, add your membership here:

Email: customer3@mail.com

Full Name: Customer Three

Phone Number: 7782354432

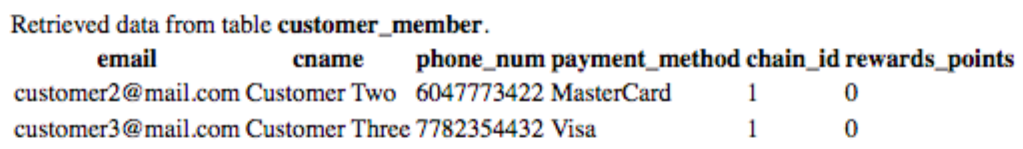
Payment Method: Visa

Theatre Chain ID: 1

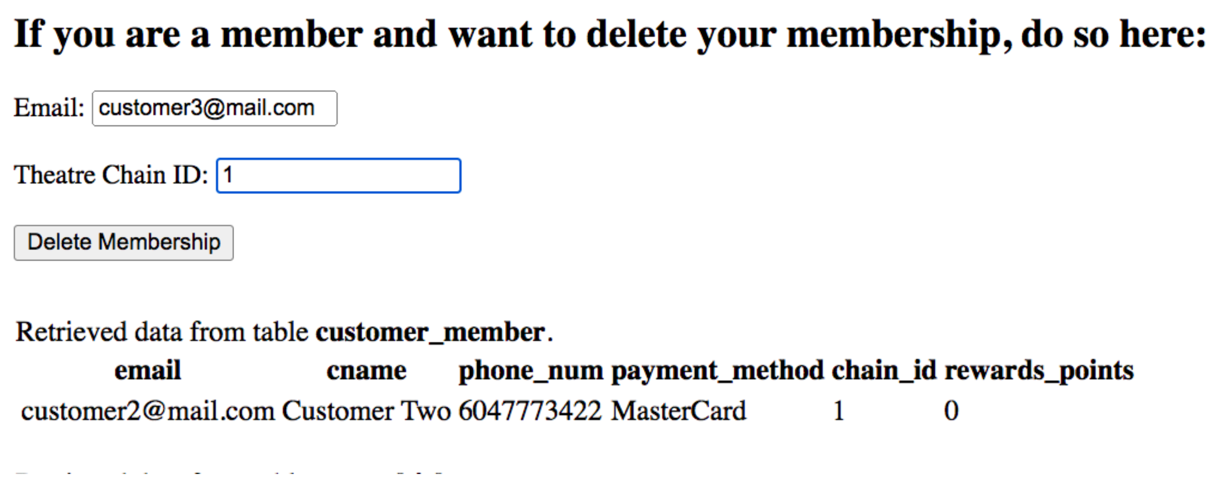
```
Retrieved data from table customer_member.
       email              cname        phone_num payment_method chain_id rewards_points
customer2@mail.com Customer Two    6047773422 MasterCard        1          0
customer3@mail.com Customer Three 7782354432 Visa              1          0
```

Queries: DELETE Operation

delete from customer_member
where email='<u>Customer3@mail.com</u>' AND chain_id = 1;



```
Retrieved data from table customer_member.
       email              cname        phone_num payment_method chain_id rewards_points
customer2@mail.com Customer Two    6047773422 MasterCard        1          0
customer3@mail.com Customer Three 7782354432 Visa              1          0
```



If you are a member and want to delete your membership, do so here:

Email: customer3@mail.com

Theatre Chain ID: 1

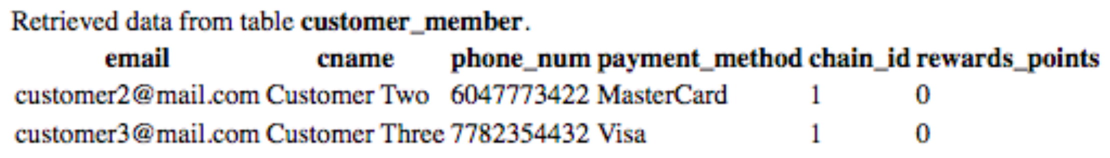
Delete Membership

Retrieved data from table **customer_member**.

email	cname	phone_num	payment_method	chain_id	rewards_points
customer2@mail.com	Customer Two	6047773422	MasterCard	1	0

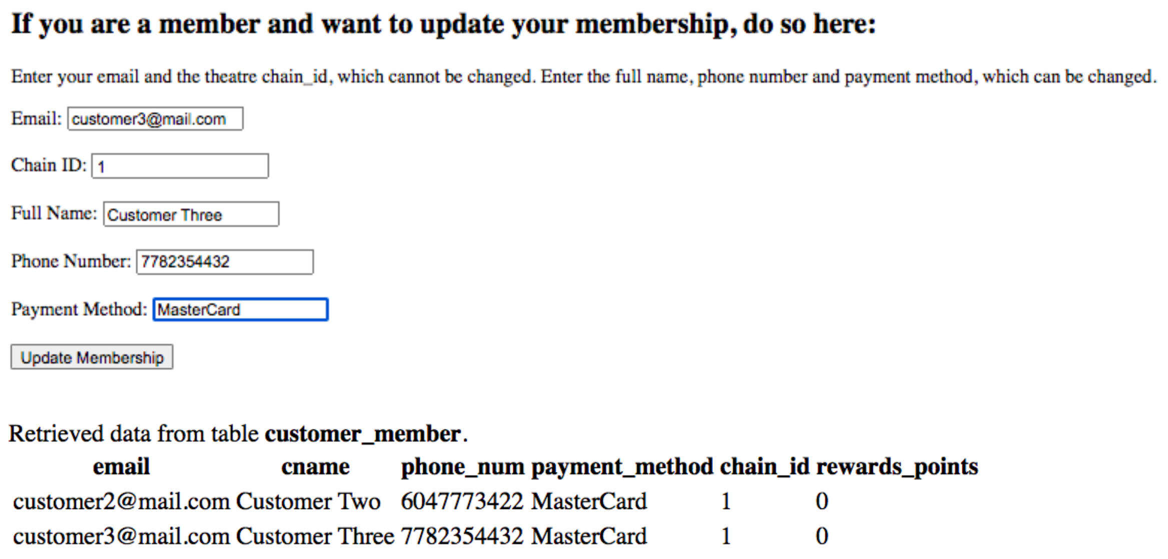
Queries: UPDATE Operation

```
update customer_nonmember
set cname= 'Customer Three', email='customer3@mail.com'
where payment_method = 'Visa';
```



Retrieved data from table **customer_member**.

email	cname	phone_num	payment_method	chain_id	rewards_points
customer2@mail.com	Customer Two	6047773422	MasterCard	1	0
customer3@mail.com	Customer Three	7782354432	Visa	1	0



If you are a member and want to update your membership, do so here:

Enter your email and the theatre chain_id, which cannot be changed. Enter the full name, phone number and payment method, which can be changed.

Email: customer3@mail.com

Chain ID: 1

Full Name: Customer Three

Phone Number: 7782354432

Payment Method: MasterCard

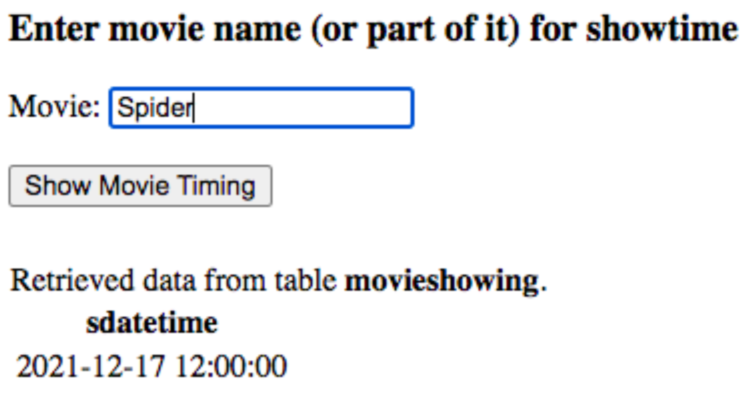
Update Membership

Retrieved data from table **customer_member**.

email	cname	phone_num	payment_method	chain_id	rewards_points
customer2@mail.com	Customer Two	6047773422	MasterCard	1	0
customer3@mail.com	Customer Three	7782354432	MasterCard	1	0

Queries: Selection

```
select sdatetime
from movieshowing
where movie_title Like '%Spider%';
```



Enter movie name (or part of it) for showtime

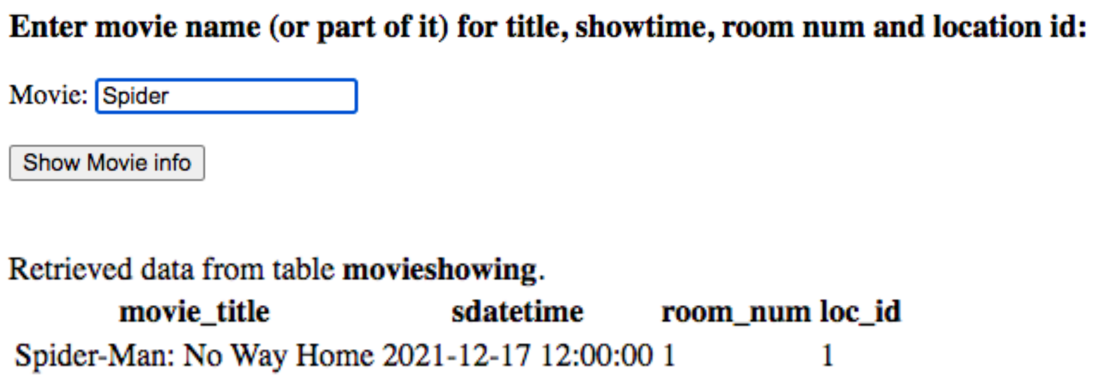
Movie: Spider

Retrieved data from table **movieshowing**.

```
sdatetime
2021-12-17 12:00:00
```

Queries: Projection

```
select sdatetime, room_num, loc_id
from movieshowing
where movie_title Like '%Spider%';
```



Enter movie name (or part of it) for title, showtime, room num and location id:

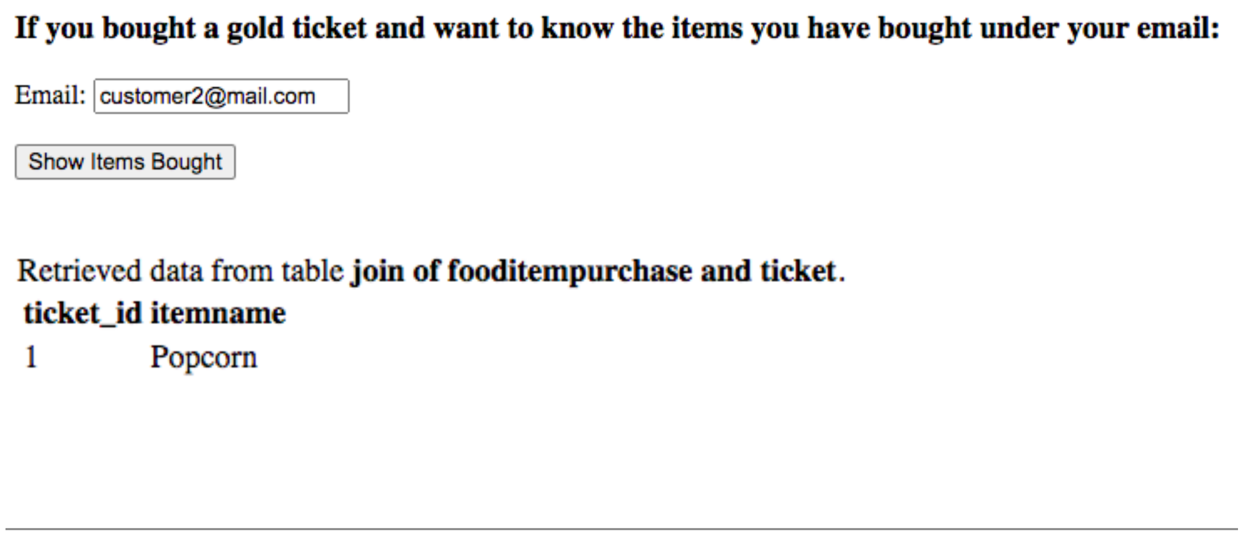
Movie: Spider

Retrieved data from table **movieshowing**.

```
movie_title                 sdatetime           room_num loc_id
Spider-Man: No Way Home 2021-12-17 12:00:00 1             1
```

Queries Join:

```
select N.ticket_id,fooditempurchase.itemname
from goldticket N
inner join fooditempurchase on N.email = fooditempurchase.email;
```



If you bought a gold ticket and want to know the items you have bought under your email:

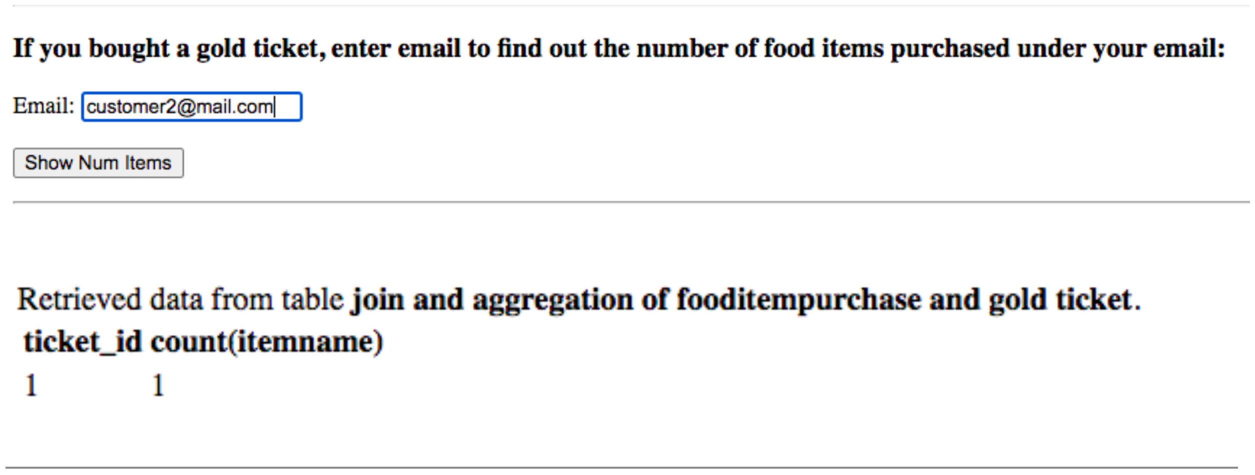
Email: customer2@mail.com

Retrieved data from table **join of fooditempurchase and ticket**.

```
ticket_id itemname
1         Popcorn
```

Queries: Aggregation group by

select ticket_id, count(sale_id)
from goldticket, fooditempurchase
where goldticket.email = fooditempurchase.email
group by ticket_id;



If you bought a gold ticket, enter email to find out the number of food items purchased under your email:

Email: customer2@mail.com

Retrieved data from table **join and aggregation of fooditempurchase and gold ticket**.

```
ticket_id count(itemname)
1         1
```

Queries: Aggregation By Having (counts the number of non rec seats at a given time and date and displays when there is greater than a particular number)

select count(nonrec_seatnum), sdatetime
from nonrec_seat
group by sdatetime
having count(nonrec_seatnum)>1;

University of British Columbia, Vancouver
Department of Computer Science

Queries: Nested Aggregation with Group By

Queries: Division

//Find all member names and emails who’ve bought every food item

SELECT m.cname, m.email FROM customer_member m WHERE NOT EXISTS ((SELECT f.itemname FROM fooditempurchase f) EXCEPT (SELECT p.itemname FROM fooditempurchase p, customer_member m WHERE p.email = m.email));

SELECT * FROM customer_member as c WHERE NOT EXISTS ((SELECT f.fooditem FROM fooditem as f) EXCEPT (SELECT p.sale_id FROM fooditempurchase p WHERE p.email = c.email));