Group 12 - Rogue One

CPSC 332

5/16/21

Quoc Luong,

Nhat Nguyen,

Gregory Pytak,

Ekaterina Yarashevich

**Introduction**

Our team was approached by a client called Force Online Marketing, an online ad marketing service that specializes in space and futuristic advertisements. We were given the task of designing a database to track their business products, store their customer information, and record orders.

**Requirements**

Business Rules

1. The online portal carries many different advertising products. We need to track each product by name, description, type, ad template, medium, cost, cost unit, as well as any product specific information.

2. Many of the different product are executed through different media partners. We need to track supplier name, address, phone, products that they supply, and cost of product. 3. Customers can only order one product in one order and can only order more than one of the same products on different orders.

4. Once an order is complete, we store customer information, including name, company, address, email address, phone number, and order information, including product purchased, date purchase, dates to execute, frequency of ad, location to run, and cost. 5. Shipping cost is determined by the customers preference. Shipping determines shipping type, cost, and delivery days to ship.

6. We need to collect payment information, including type, number , description, expiration and verification code.

7. A customer can have more than one payment type.

8. Make sure that the Customer ID is the first 2 letter of his first name followed by number. Eg. If the Customer’s name is Steve Gates, then his Customer ID can be ST2048.

9. The system should track campaigns purchased based on the product ordered, the date range to run the campaign, whether its active, the countries the ad should be ran, the media it will be ran on, and any relevant information.

10. The system will also track campaign analytics based on impressions (where they displayed) every hour. For each impression, track the online URL location appeared, the media it is displayed on, the site it is displayed, the number of impressions an hour, the number of impressions on that site per hour, the number of clicks per impression, and the date and hour of the impression.

Additional Requirements

1. One SCRIPT to create this database (call it ForceAds) with MySQL server.

2. A site is collection information about one of our video ad products on Youtube. They like to promote a new marketing opportunity for previous customers that the utilized the product in the past year for a promotional deal. For this purpose, create a VIEW that finds the names, company and phone numbers of all customers who purchased the product along with the total spend on the product the past year.

3. Create a view which has first names and last names of all members who purchased the mass market via email.

4. Create a view which shows the product name and type of all products that sold more than 25 units.

5. Modify the view created in Q4 to show the product name and manufacturer of all products that sold more than 25 units and generated more than $200 revenue.

6. Create trigger on the Products so that every time a purchase payment total reaches $2,000, a new entry is made in the Promotion Qualitied table to list the customers eligible to receive special offers. The qualified table will have the following (Hint-The trigger will be on Product table).

a. Customer Name

b. Customer Email

c. Customer Phone

d. Amount Spent

e. Action (indicate Qualified)

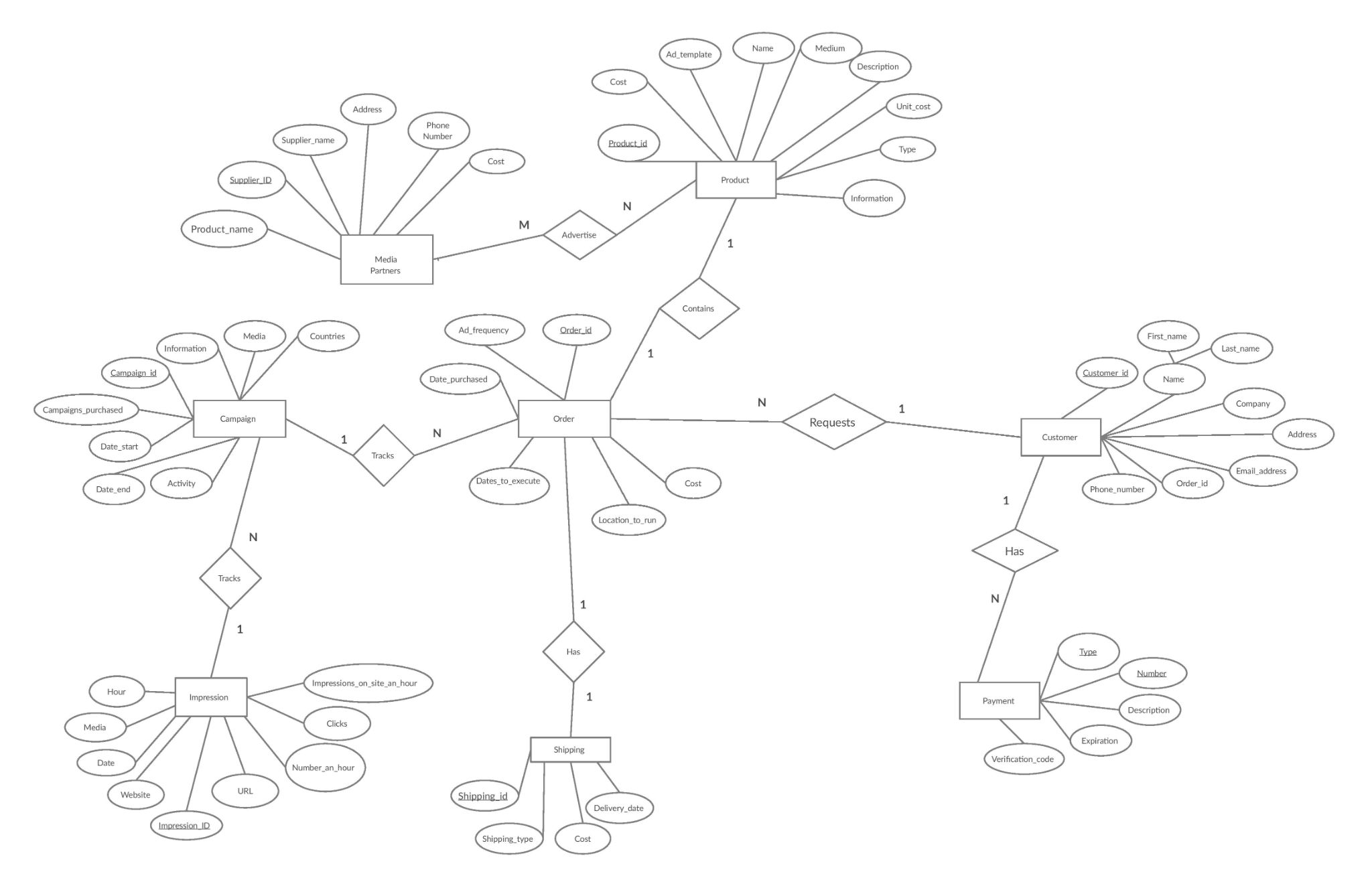
f. Date of Submission

7. Create a script to do the following (Write the script for this)

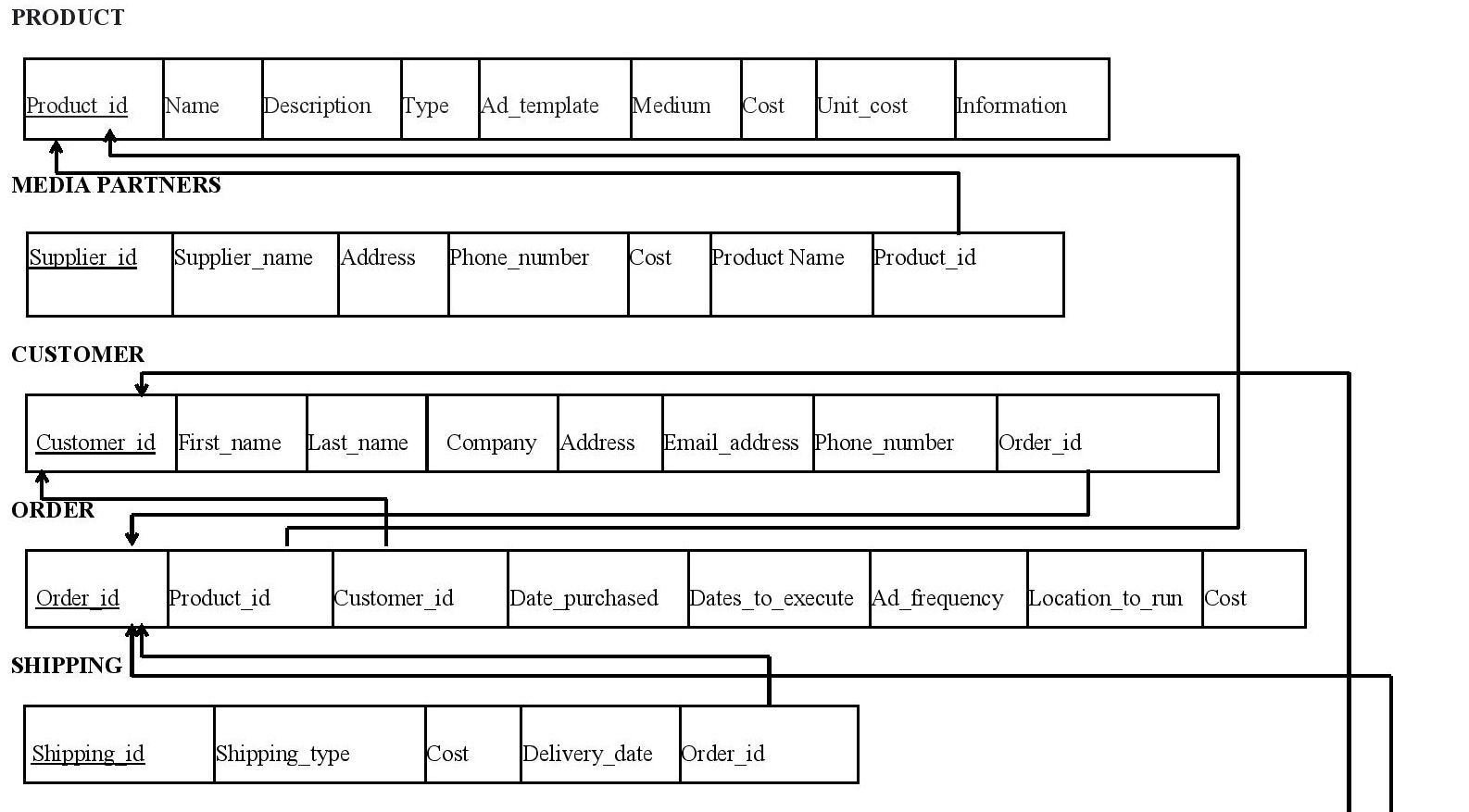
a. If first time backup take backup of all the tables

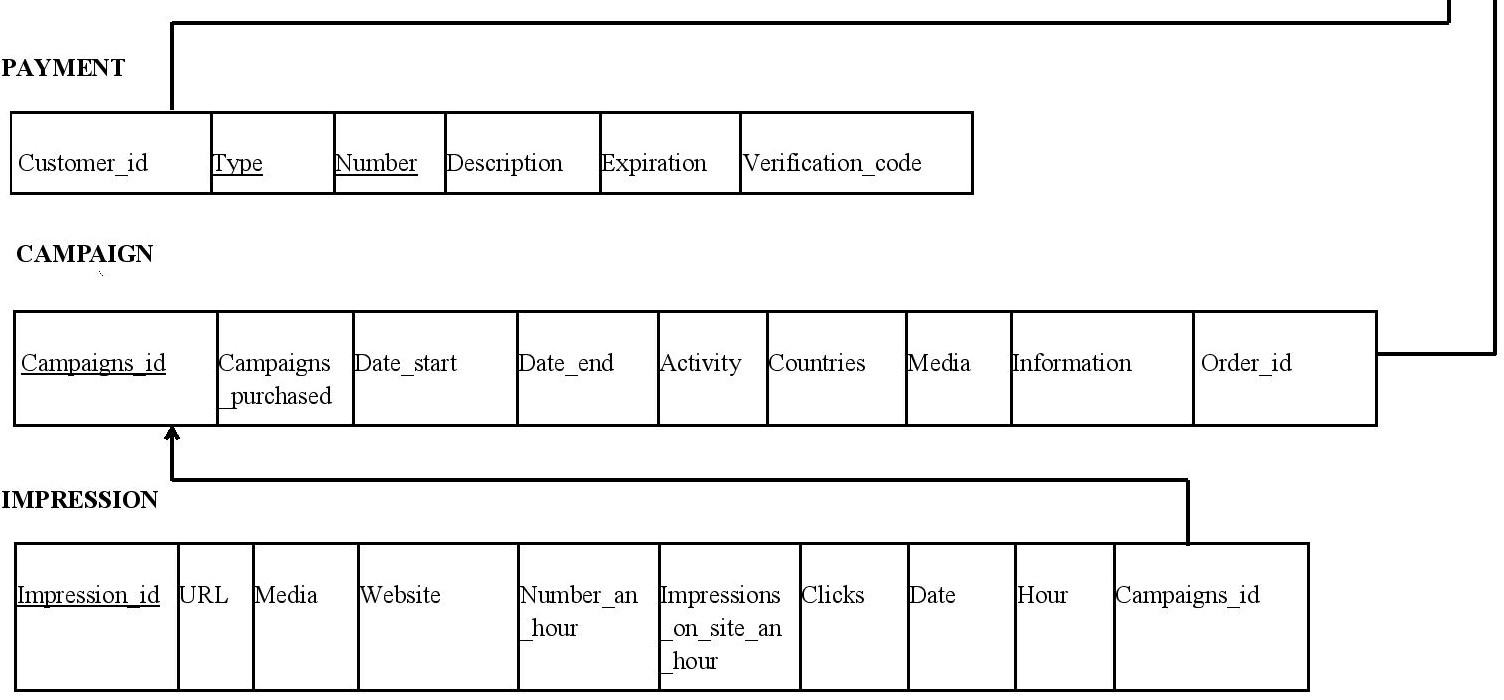
b. If not the first time remove the previous backup tables and take new backups.

**ER Diagram**

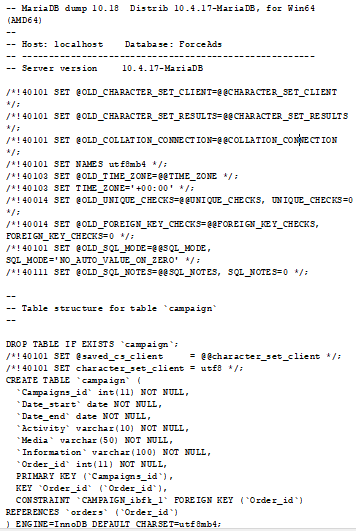


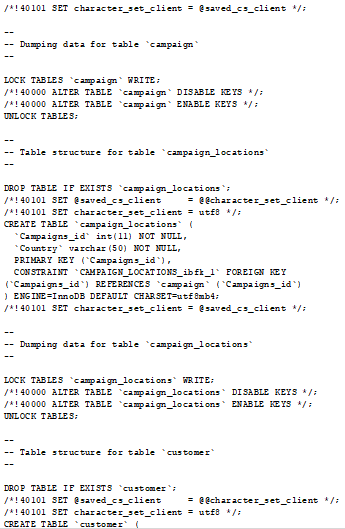
**Relational Model**

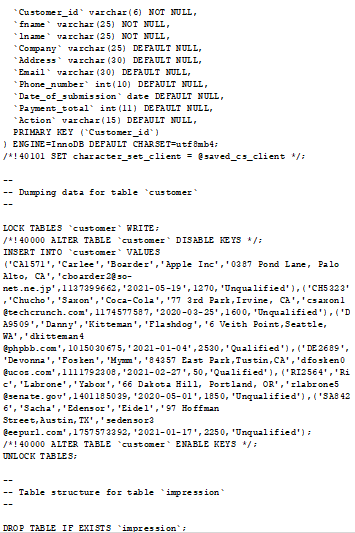


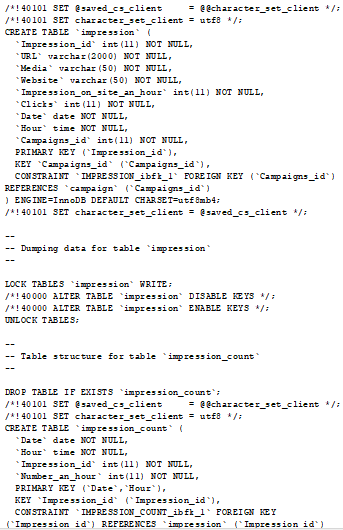


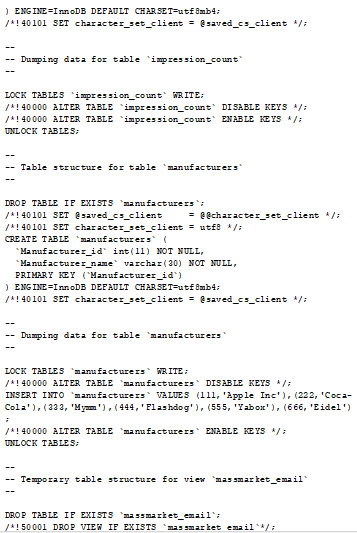
**Physical Model**

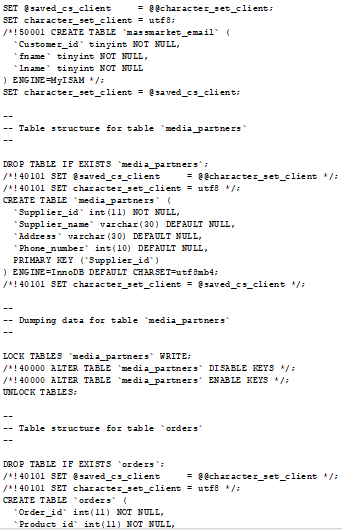


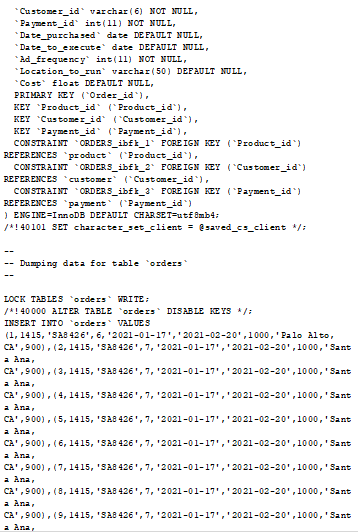


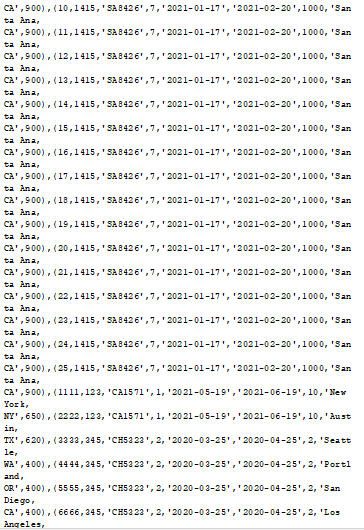
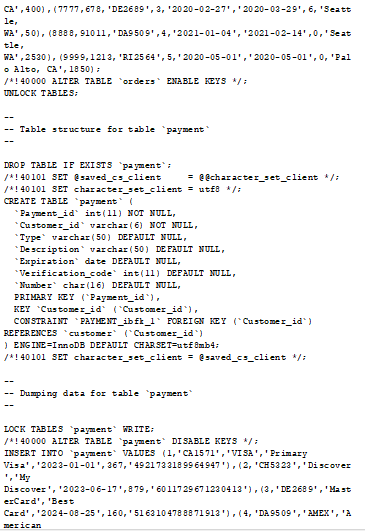


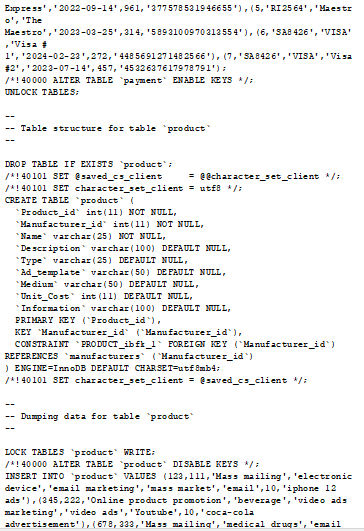


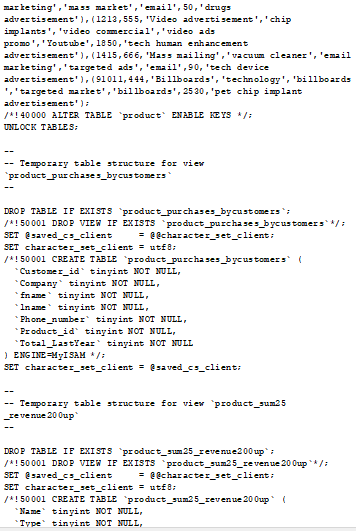


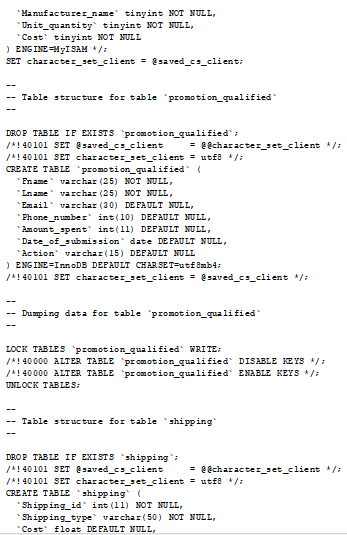


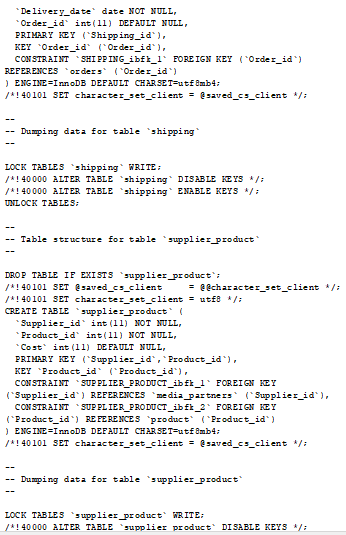


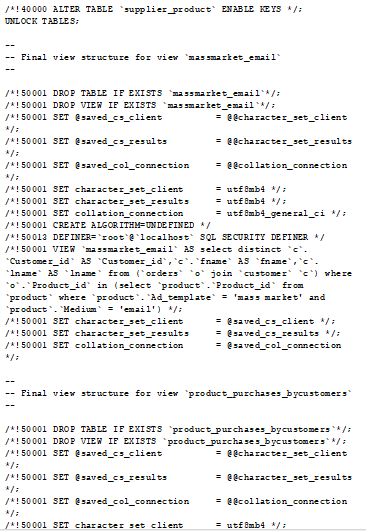
  


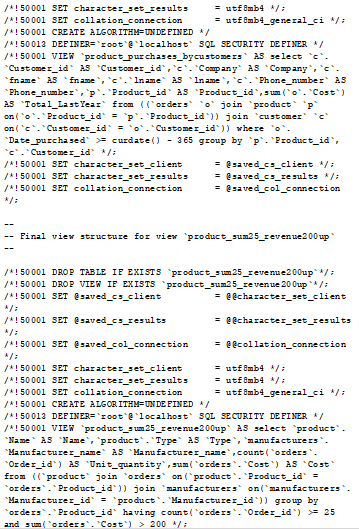


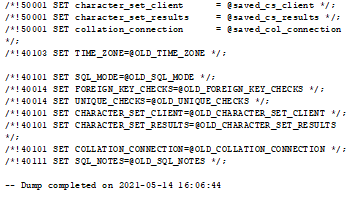












**Overview**

Beginning with the requirements, we made a relational model with the data given to us from the business rules. Instructions involving possible tables, such as a Customer, were given attributes to add onto it. Some instructions became guidelines which were of use later on such as how to format the customer’s ID and the amount of payment types a customer can have. Using the relational model we formed an ER diagram where we placed each table as an entity with its relating attributes, then connected each entity that possessed a relationship to one another.

From there we moved onto the additional requirements which required us to create a website and some SQL queries. The site is used to collect information based on customers who owned a previous product. The SQL queries include views and a trigger which are in design based on the customer information as well as a script to backup new tables and override old backups. Once the code was designed and finished on MySQL, it was made into a dump file and moved into the site.

For the purpose of demonstrating our ForceAds database, we made a simple website front end using HTML, CSS, Bootstrap, and php with back end using xampp Apache. First, we created the scripts.js file with a simple bootstrap designed by other developers. It’s an open source so we can download without having issues with copyright. Then we changed some of its features including the header, all the tabs name including the contacts, about, services, and portfolio. Also, we wrote some queries in queries.sql to get the data then we created queries\_functions.php to display the result as view.

**SQL Queries**

DROP DATABASE IF EXISTS ForceAds;

CREATE DATABASE ForceAds;

USE ForceAds;

DROP TABLE IF EXISTS PRODUCT;

CREATE TABLE PRODUCT (

Product\_id INT NOT NULL,

Name VARCHAR(25) NOT NULL,

Description VARCHAR(100) ,

Type VARCHAR(25),

Ad\_template VARCHAR(50),

Medium VARCHAR(50) ,

Cost FLOAT,

Unit\_Cost INT,

Information VARCHAR(100),

Manufacturer\_id INT NOT NULL,

PRIMARY KEY (Product\_id),

FOREIGN KEY(Manufacturer\_id) REFERENCE MANUFACTURERS

);

DROP TABLE IF EXISTS MANUFACTURERS;

CREATE TABLE MANUFACTURERS(

Manufacturer\_id INT NOT NULL,

Manufacturer\_name VARCHAR(30) NOT NULL,

PRIMARY KEY(Manufacturer\_id)

);

DROP TABLE IF EXISTS CUSTOMER;

CREATE TABLE CUSTOMER(

Customer\_id VARCHAR(6) NOT NULL,

fname VARCHAR(25) NOT NULL,

lname VARCHAR(25) NOT NULL,

Company VARCHAR(25),

Address VARCHAR30),

Email VARCHAR(30),

Phone\_number INT(10),

Payment\_total INT(11),

Date\_of\_submission DATE,

Action VARCHAR(15),

PRIMARY KEY(Customer\_id)

);

DROP TABLE IF EXISTS PROMOTION QUALIFIED;

CREATE TABLE PROMOTION QUALIFIED(

fname VARCHAR(25) NOT NULL,

lname VARCHAR(25) NOT NULL,

Email VARCHAR(30),

Phone\_number INT(10),

Amount\_spent INT,

Date\_of\_submission DATE,

Action VARCHAR(15)

);

DROP TABLE IF EXISTS ORDER;

CREATE TABLE ORDER(

Order\_id INT NOT NULL,

Product\_id INT NOT NULL,

Customer\_id INT NOT NULL,

Payment\_id INT NOT NULL,

Date\_purchased DATE,

Date\_to\_execute DATE,

Ad\_frequency INT NOT NULL,

Location\_to\_run VARCHAR(50),

Cost FLOAT,

PRIMARY KEY(Order\_id),

FOREIGN KEY (Product\_id) REFERENCE PRODUCT,

FOREIGN KEY(Customer\_id) REFERENCE CUSTOMER,

FOREIGN KEY(Payment\_id) REFERENCE PAYMENT

);

DROP TABLE IF EXISTS SHIPPING;

CREATE TABLE SHIPPING(

Shipping\_id INT NOT NULL,

Shipping\_type VARCHAR(50) NOT NULL,

Cost FLOAT ,

Delivery\_date DATE NOT NULL,

Order\_id INT,

PRIMARY KEY(Shipping\_id),

FOREIGN KEY(Order\_id) REFERENCE ORDER

);

DROP TABLE IF EXISTS PAYMENT;

CREATE TABLE PAYMENT(

Payment\_id INT NOT NULL,

Number INT NOT NULL,

Customer\_id INT NOT NULL,

Type VARCHAR(50),

Description VARCHAR(50),

Expiration DATE,

Verification\_code INT,

FOREIGN KEY(Customer\_id) REFERENCE CUSTOMER,

PRIMARY KEY(Payment\_id)

);

DROP TABLE IF EXISTS CAMPAIGN;

CREATE TABLE CAMPAIGN(

Campaigns\_id INT NOT NULL,

Date\_start DATE NOT NULL,

Date\_end DATE NOT NULL,

Activity VARCHAR(10) NOT NULL,

Media VARCHAR(50) NOT NULL,

Information VARCHAR(100) NOT NULL,

Order\_id INT NOT NULL,

PRIMARY KEY(Campaigns\_id),

FOREIGN KEY(Order\_id) REFERENCE ORDER

);

DROP TABLE IF EXISTS CAMPAIGN\_LOCATIONS;

CREATE TABLE CAMPAIGN\_LOCATIONS(

Campaigns\_id INT NOT NULL,

Country VARCHAR(50) NOT NULL,

PRIMARY KEY(Campaigns\_id),

FOREIGN KEY(Campaigns\_id) REFERENCE CAMPAIGN

);

DROP TABLE IF EXISTS IMPRESSION;

CREATE TABLE IMPRESSION(

Impression\_id INT NOT NULL

URL VARCHAR(2000) NOT NULL,

Media VARCHAR(50) NOT NULL,

Website VARCHAR(50) NOT NULL,

Impression\_on\_site\_an\_hour INT NOT NULL,

Clicks INT NOT NULL,

Date DATE NOT NULL,

Hour TIME NOT NULL,

Campaigns\_id INT NOT NULL,

PRIMARY KEY(Impression\_id),

FOREIGN KEY(Campaigns\_id) REFERENCE CAMPAIGN

);

DROP TABLE IF EXISTS IMPRESSION\_COUNT;

CREATE TABLE IMPRESSION\_COUNT(

Date DATE NOT NULL,

Hour TIME NOT NULL,

Impression\_id INT NOT NULL,

Number\_an\_hour INT NOT NULL,

PRIMARY KEY(Impression\_id),

PRIMARY KEY(Date),

PRIMARY KEY(Hour),

FOREIGN KEY(Impression\_id) REFERENCE IMPRESSION

);

DROP TABLE IF EXISTS MEDIA\_PARTNERS;

CREATE TABLE MEDIA\_PARTNERS(

Supplier\_id INT NOT NULL,

Supplier\_name VARCHAR(30),

Address VARCHAR(30),

Phone\_number INT(10),

PRIMARY KEY(Supplier\_id)

);

DROP TABLE IF EXISTS SUPPLIER\_PRODUCT;

CREATE TABLE SUPPLIER\_PRODUCT(

Supplier\_id int NOT NULL,

Product\_id int NOT NULL,

Cost int.

PRIMARY KEY(Supplier\_id),

PRIMARY KEY(Product\_id)

FOREIGN KEY(Supplier\_id) REFERENCE MEDIA\_PARTNERS

FOREIGN KEY(Product\_id) REFERENCE PRODUCT

);

delimiter //

CREATE TRIGGER special\_offer AFTER INSERT ON Customer FOR EACH ROW

BEGIN IF NEW.payment\_total >= 2000 THEN

INSERT INTO PROMOTION\_QUALIFIED SET

fname = NEW.fname, lname = NEW.lname, email = NEW.email, phone\_number = NEW.phone\_number, amount\_spent = NEW.payment\_total, action = NEW.action, date\_of\_submission = NEW.date\_of\_submission;

END IF;

END;

//

delimiter ;

delimiter //

CREATE PROCEDURE Product\_Fullerton()

BEGIN

SELECT Product.Name, Product.Type, Product.Medium, COUNT(Orders.Customer\_id) FROM PRODUCT, ORDERS WHERE (Location\_to\_run = “Fullerton” AND ORDERS.Product\_id = PRODUCT.Product\_id);

END;

//

delimiter ;

--views

--requirement#2

CREATE VIEW PRODUCT\_PURCHASES\_byCUSTOMERS AS SELECT C.Customer\_id, C.Company, C.fname, C.lname, C.Phone\_number, P.Product\_id, SUM(O.Cost) as Total\_LastYear FROM (ORDERS AS O JOIN PRODUCT AS P ON O.Product\_id=P.Product\_id JOIN CUSTOMER AS C ON C.Customer\_id=O.Customer\_id) WHERE O.Date\_purchased>= CURDATE()-365 GROUP BY P.Product\_id, C.Customer\_id;

--requirement#3

CREATE VIEW MASSMARKET\_EMAIL AS SELECT DISTINCT C.Customer\_id, fname,lname from (ORDERS as O JOIN CUSTOMER AS C) where Product\_id IN(SELECT Product\_id from PRODUCT WHERE Ad\_template ='mass market' AND Medium = 'email');

--requirement#4 and 5

CREATE VIEW PRODUCT\_SUM25\_REVENUE200UP AS SELECT PRODUCT.Name, PRODUCT.Type, Manufacturer\_name, COUNT(ORDERS.Order\_id) AS Unit\_quantity, SUM(ORDERS.Cost) as Cost FROM ((PRODUCT INNER JOIN ORDERS ON PRODUCT.Product\_id = ORDERS.Product\_id)

INNER JOIN MANUFACTURERS ON MANUFACTURERS.Manufacturer\_id = PRODUCT.Manufacturer\_id)

GROUP BY ORDERS.Product\_id

HAVING COUNT(ORDERS.Order\_id) >= 25 AND SUM(ORDERS.Cost) > 200;

DROP TABLE IF EXISTS `CAMPAIGN\_BACKUP`;

CREATE TABLE `CAMPAIGN\_BACKUP` LIKE `CAMPAIGN`;

INSERT INTO `CAMPAIGN\_BACKUP` SELECT \* FROM `CAMPAIGN`;

DROP TABLE IF EXISTS `CAMPAIGN\_LOCATIONS\_BACKUP`;

CREATE TABLE `CAMPAIGN\_LOCATIONS\_BACKUP` LIKE `CAMPAIGN\_LOCATIONS`;

INSERT INTO `CAMPAIGN\_LOCATIONS\_BACKUP` SELECT \* FROM `CAMPAIGN\_LOCATIONS`;

DROP TABLE IF EXISTS `CUSTOMER\_BACKUP`;

CREATE TABLE `CUSTOMER\_BACKUP` LIKE `CUSTOMER`;

INSERT INTO `CUSTOMER\_BACKUP` SELECT \* FROM `CUSTOMER`;

DROP TABLE IF EXISTS `IMPRESSION\_BACKUP`;

CREATE TABLE `IMPRESSION\_BACKUP` LIKE `IMPRESSION`;

INSERT INTO `IMPRESSION\_BACKUP` SELECT \* FROM `IMPRESSION`;

DROP TABLE IF EXISTS `IMPRESSION\_COUNT\_BACKUP`;

CREATE TABLE `IMPRESSION\_COUNT\_BACKUP` LIKE `IMPRESSION\_COUNT`;

INSERT INTO `IMPRESSION\_COUNT\_BACKUP` SELECT \* FROM `IMPRESSION\_COUNT`;

DROP TABLE IF EXISTS `MANUFACTURERS\_BACKUP`;

CREATE TABLE `MANUFACTURERS\_BACKUP` LIKE `MANUFACTURERS`;

INSERT INTO `MANUFACTURERS\_BACKUP` SELECT \* FROM `MANUFACTURERS`;

DROP TABLE IF EXISTS `MEDIA\_PARTNERS\_BACKUP`;

CREATE TABLE `MEDIA\_PARTNERS\_BACKUP` LIKE `MEDIA\_PARTNERS`;

INSERT INTO `MEDIA\_PARTNERS\_BACKUP` SELECT \* FROM `MEDIA\_PARTNERS`;

DROP TABLE IF EXISTS `ORDERS\_BACKUP`;

CREATE TABLE `ORDERS\_BACKUP` LIKE `ORDERS`;

INSERT INTO `ORDERS\_BACKUP` SELECT \* FROM `ORDERS`;

DROP TABLE IF EXISTS `PAYMENT\_BACKUP`;

CREATE TABLE `PAYMENT\_BACKUP` LIKE `PAYMENT`;

INSERT INTO `PAYMENT\_BACKUP` SELECT \* FROM `PAYMENT`;

DROP TABLE IF EXISTS `PRODUCT\_BACKUP`;

CREATE TABLE `PRODUCT\_BACKUP` LIKE `PRODUCT`;

INSERT INTO `PRODUCT\_BACKUP` SELECT \* FROM `PRODUCT`;

DROP TABLE IF EXISTS `PROMOTION\_QUALIFIED\_BACKUP`;

CREATE TABLE `PROMOTION\_QUALIFIED\_BACKUP` LIKE `PROMOTION\_QUALIFIED`;

INSERT INTO `PROMOTION\_QUALIFIED\_BACKUP` SELECT \* FROM `PROMOTION\_QUALIFIED`;

DROP TABLE IF EXISTS `SHIPPING\_BACKUP`;

CREATE TABLE `SHIPPING\_BACKUP` LIKE `SHIPPING`;

INSERT INTO `SHIPPING\_BACKUP` SELECT \* FROM `SHIPPING`;

DROP TABLE IF EXISTS `SUPPLIER\_PRODUCT\_BACKUP`;

CREATE TABLE `SUPPLIER\_PRODUCT\_BACKUP` LIKE `SUPPLIER\_PRODUCT`;

INSERT INTO `SUPPLIER\_PRODUCT\_BACKUP` SELECT \* FROM `SUPPLIER\_PRODUCT`;

**Final Product**

The source code of website can be viewed in <https://github.com/quocluong32/CSPC322_Project.git>

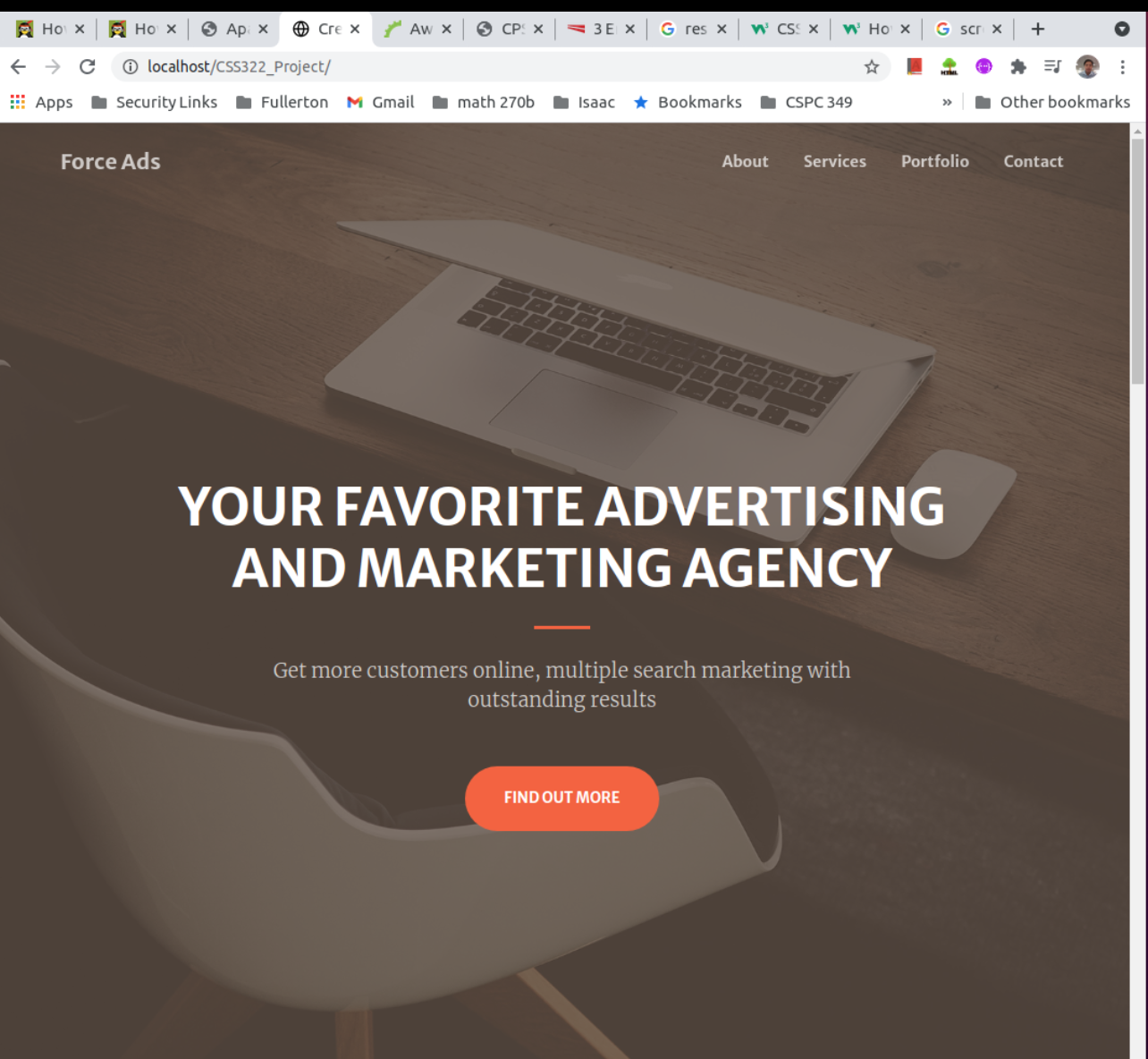


Figure1: our first page when running xampp apache.

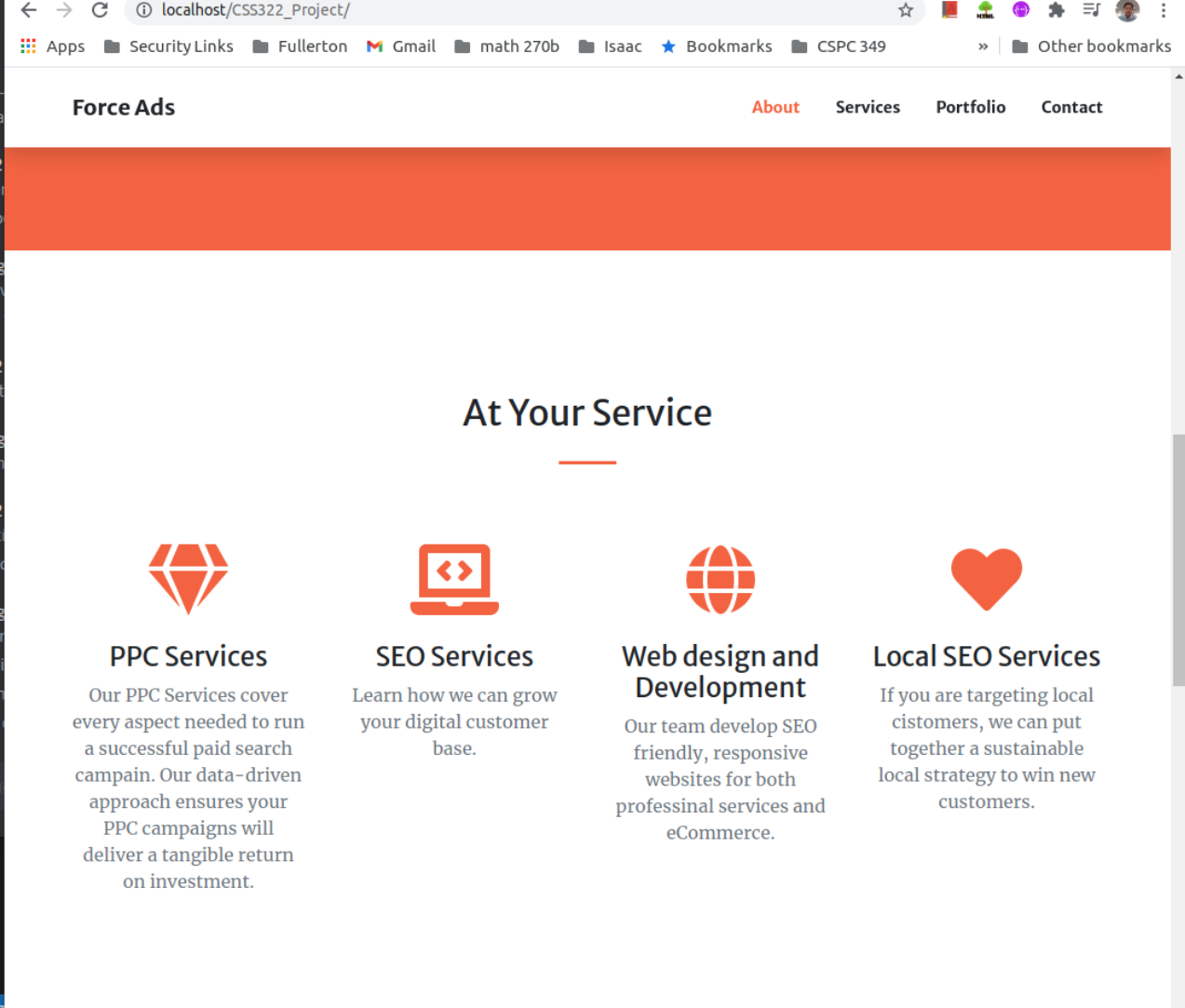


Figure 2: The About tab describes our service.

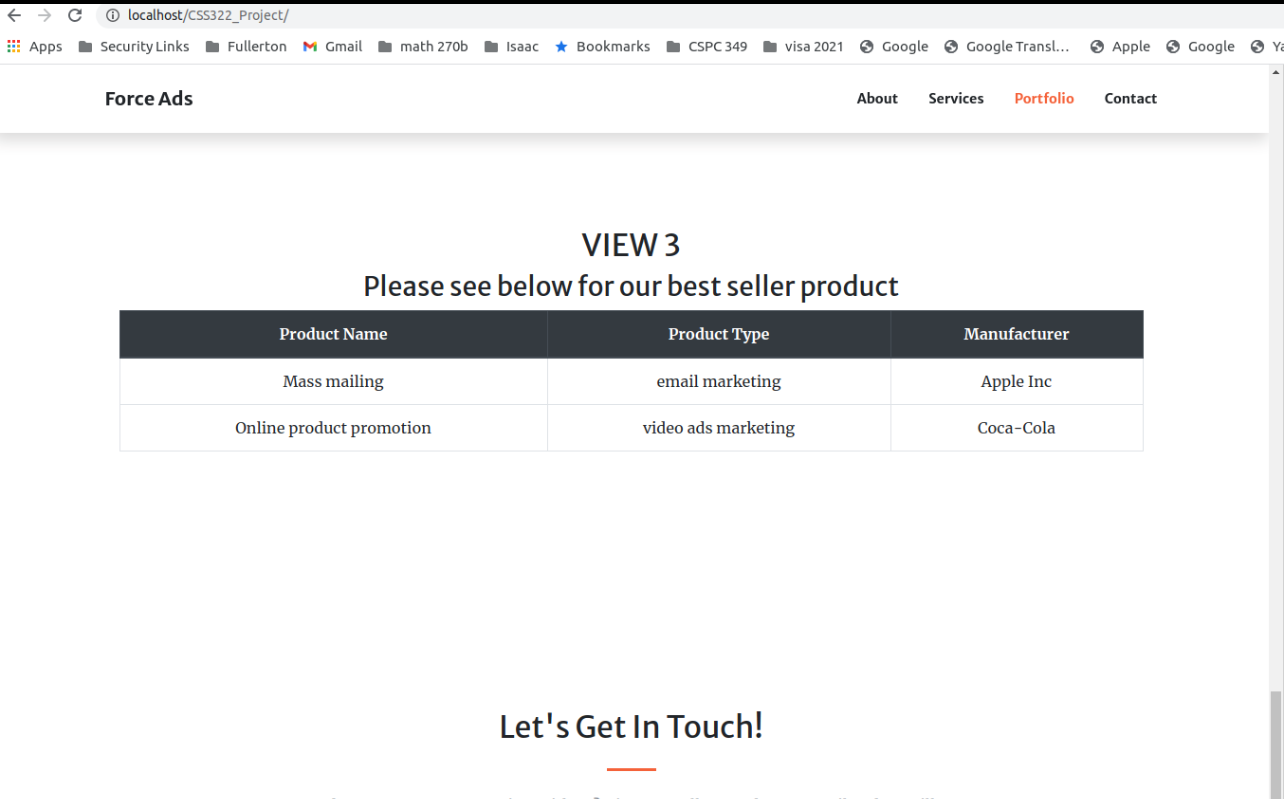


Figure 3: A view shows the product name, product type and manufacturer of all products that sold more than 25 units and generated more than $200 revenues.

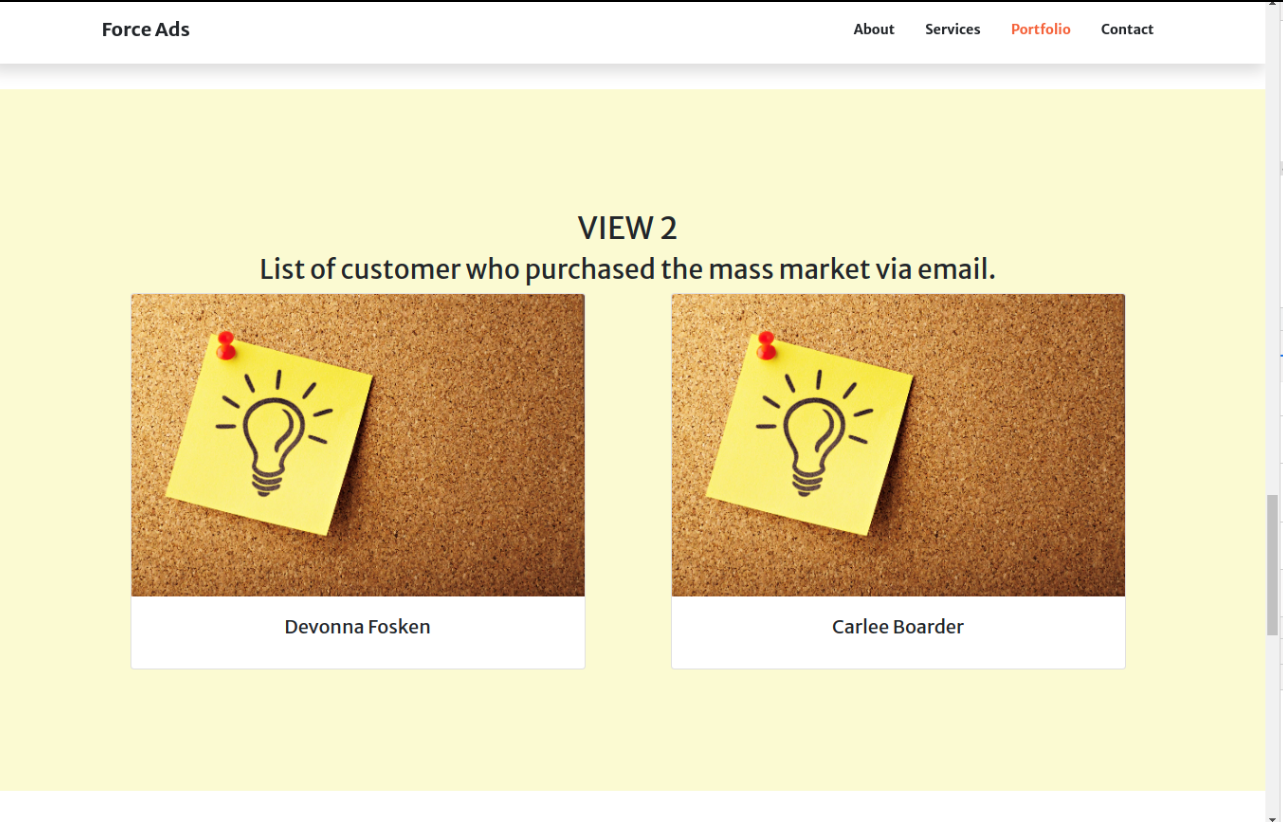


Figure 4:A view which has first names and last name of all members who purchased the mass market.

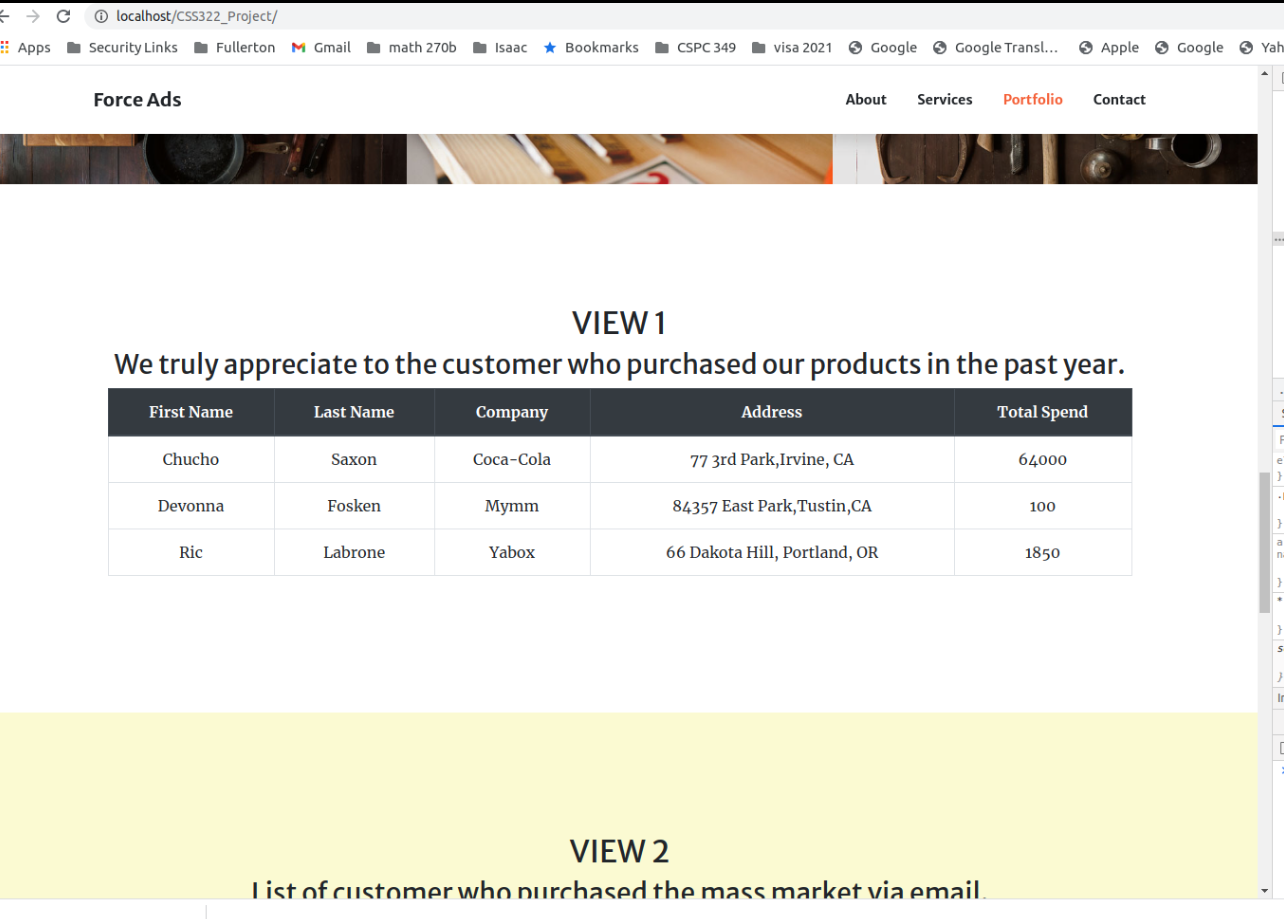


Figure 5: A view shows names, companies, and phone numbers of customers purchased last year.

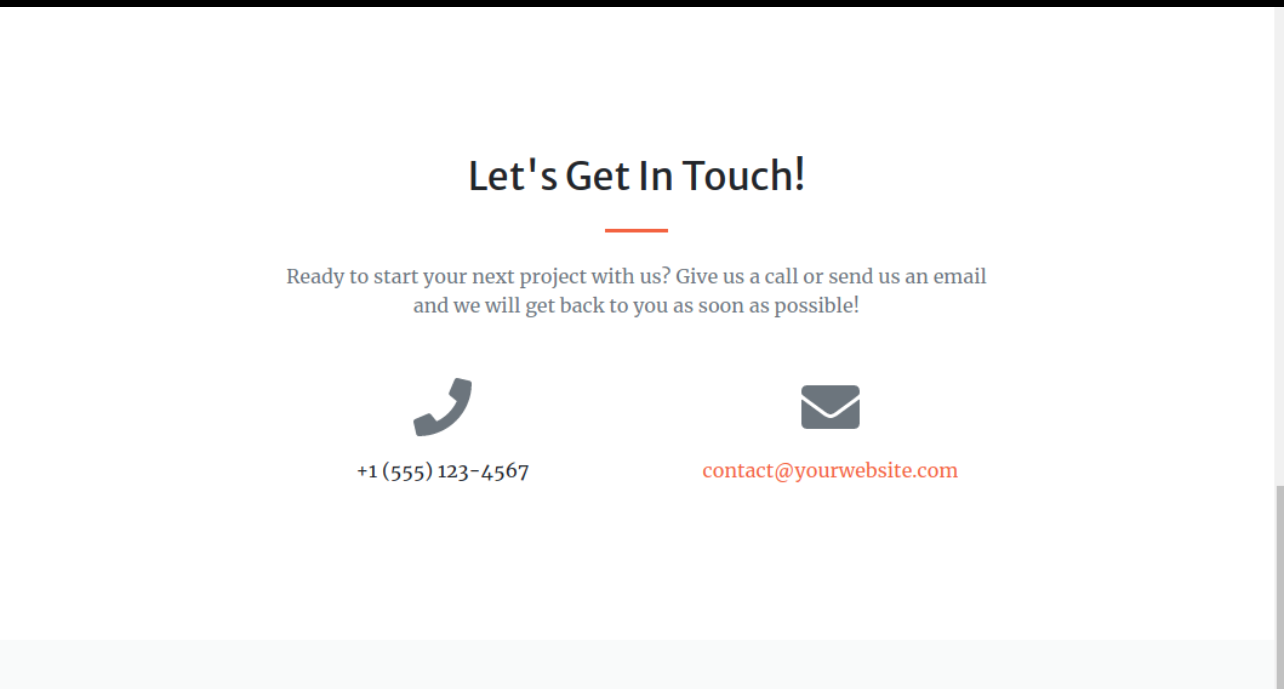


Figure 6: contact tab with phone number and email.

We can only reach the stage of displaying queries and changing some basic features of our website including contacts, about, etc.

**Summary**

Our team was approached by Force Online Marketing, a company which tasked us in creating an online marketing service for space and futuristic advertisements. Our job was to create a database to track their products, customer information, and orders. The database we have designed, named ForceAds, is able to store all information needed for an online marketing service. As well, we created a website used to access the database which is user friendly.

**Team Review**

Quoc Luong:

The project overall is challenging, since we have to deal with practical business logic rules, and a lot of things that all of us are not familiar with like how to generate mock data, as well as PHP (back-end programming language use for this project). Other than that, my daughter was in the hospital for 3 days the last 2 weeks that made me behind in everything. However, we’ve worked as a team to overcome all the challenges. I also received help from my teammates to allow my absences for a couple of group’s meetings. Regarding my contribution to the project, I help to review the queries to create views, and write codes to create websites utilizing HTML, Bootstrap and PHP. All of the site functions work as expected. The greatest benefit for completing the project is I know the process of how to create the database from the beginning to the end, and learning the new programming language PHP that will give me a head start to continue to learn wordpress in the summer.

Gregory Pytak:

The project was simple to start out with in creating the relational model and the ER diagram as we have had experience creating them before. Once we moved onto the additional requirements with the script, trigger, views, and website then it quickly became more challenging. The script we originally created had to go through many edits as we found out what tables and attributes we needed to keep, delete, or add, but this is something I would expect from any development process. After much editing and testing we did end up with a functioning database and website. This was one of the most challenging group assignments I have worked on due solely to the material itself. Overall, it feels satisfying to see what was created from it and I think it was a great example as to what real web development was like.

Nhat Nguyen:

The project was a real challenge for me. There are many tasks we have to complete in a couple of weeks. Through time, we had to come up with different drafts of the relational model and the ER diagram. We had the first draft of the relational model and the ER diagram. Then, we wrote the script to create the database and move on to the trigger, views, and website. Along with those tasks, we had to do the normalization and changed most of the structure in our design a couple of times until we were satisfied with it. Scheduling times for everyone to meet is also a challenge because everyone has a different schedule.However, we managed to meet once every week to distribute tasks for every one, and to update the script and the design. I know more about how to translate from the ER diagram and relational model to the sql script, and create the backup script. I tried to do a little self-studying about the js, php, and bootstrap and Quoc Luong helped me a lot to understand how to set up the Apache back end on Xampp, however, I was not able to run the html on my end so he had to take care most of the website task. Overall, it was a fun experience to work as a team to build something from scratch. After this project,I enjoy learning more about web front end.

Ekaterina Yarashevich:

The project was a very immersive experience in understanding how relational databases work and how to use SQL efficiently as a tool for it. The initial phase of the project was designing an ER diagram and a Relational model which was an excellent opportunity to practice teamwork skills while we were discussing aspects of different entities and their attributes and how they connect to each other. The normalization of our initial design helped me to process better how to represent the design better and make it more functional in the actual physical model of it. The most unexpected challenging part of the project was the process of coming up with data tuples to populate the tables, in order to test the database requirements, since it was when I got real glimpse of how actual tuples connect to each other in a relational database and how built in functionality works for Primary keys and data entries relate to each other in different tables and how relationships between them implemented through foreign keys work. The views implementation helped to experience JOIN functionality in SQL and recognize how CREATE VIEW method allows to create virtual tables for easy access to the particular data. The extra credit implementation of the website demonstrated the end product of what a development of a full scale product with a simple database will look like on the customer side. The team work involved in building this project trained better communication and organizational skills in me and gave an insight of what a potential work environment will look like in a tech industry.