

Simple Kitchen



From Our Kitchen to Yours Family Table

CONTENTS

Summary	2
Stakeholders	3
Business Rules	4
Glossary	4
Data Questions	5
Conceptual Model	6
Logical Model	7
Physical Database Design	8
Data Creation	13
Data Manipulation and Answering Data Question	24
Implementation	28
Reflection	31
Summary	32

SUMMARY

Pandemic has changed the way how people shop for holiday meals. With the uptick of Covid19 and flu infections in the upcoming Holiday seasons, grocery shopping becomes a health hazard activity and a stressful one. Simple kitchen is a meal prep delivery business located in Richmond, California, and has been operating as a pop-up kitchen since March 2020. As a pop-up kitchen, Simple Kitchen has offered weekly healthy meal prep services for the Bay Area's busy family. Their customer orders are stored on Instagram and Facebook messages, emails, and occasionally a spreadsheet. Thus, this is not an effective way of maintaining data on customers and orders. Simple kitchen has a massive following on several social media sites and would like to expand their business model from just a meal prep service. Several customers reached out to ask if Simple Kitchen offers a holiday meal delivery service with meal prep instruction. Simple kitchen found that the lack of a database system hindered them from

improving their little pop-up business to a full-fledged business. Instant Database Solution is hired to build a database for Simple Kitchen to streamline their ordering process.

Instant Database Solution has conducted a series of meetings with the Simple Kitchen owner and staff to understand their business practices and bottlenecks in their current business process. An initial draft of the conceptual model has identified the entities, attributes, and relationships between the customer information, ordering, delivery scheduling, billing, and recipe contents. The goals of Instant Database Solution at this database design stage are to prototype with a conceptual and logical model. Then proceed to start a physical database design, data creation, data manipulation, and implementation to answer business questions. A prototype database will test out as well as plan for iteration activities to optimize the database.

STAKEHOLDERS

This project's stakeholders are the Simple Kitchen's owners, meal prep staff, social media staff, and loyal customers. Instant Database Solution has proposed a comprehensive database to streamline meal selection via dietary choice, offer holiday menu, schedule delivery, manage recipe content, and order, which will improve Simple Kitchen workflow process significantly and profitability. Owners will have easy access to the database for the customer list, business accounting, future meal planning, and marketing strategy. Simultaneously, the meal prep staff will have immediate access to the upcoming meal orders, delivery schedule, and order status. Customers will have an account to put a new order and review previous orders. Lastly, social media staff can track which meal is popular among the customers in the database to improve followers' engagement and increase the number of followers.

BUSINESS RULES

- A customer selects a dietary choice.
- A customer picks a holiday menu.
- A customer picks one or many meals.
- One meal has one or many ingredients.
- A customer places an order.
- An order contains one or more ingredient details.
- An invoice payment generates to pay.
- Staff prep the meal order.
- A staff completes an order.
- A staff delivers the order to the customer address.
- An order contains one or more detailed recipes.
- A recipe includes a website for cooking instruction.

GLOSSARY

A **customer** is a person who orders a Simple Kitchen meal prep service.

Delivery is a service that Simple Kitchen offers to deliver meals from their kitchen to customer's homes.

Dietary consists of special diet preferences for a customer who wants a vegetarian, vegan, paleo, healthy, gluten-free, or allergy-free meal.

Holiday Menu consists of the upcoming Thanksgiving, Hanukka, Christmas, and New Year dinners.

The recipe consists of the **ingredients** for the meal prep and the cost of each ingredient to prep.

Invoice is a bill that Simple Kitchen provides the customer.

The **meal type** consists of breakfast, lunch, dinner, appetizer, and cocktail.

Meal Item is an identified meal-type and holiday menu.

Order contains information on the customer, the meal order, staff who prepare, and

Order Item contains information on the recipe and meal item to streamline to the order.

Payment is how the customer will pay via debit card, PayPal, or Apple pay.

Quantity is one or many meals that the customer will order.

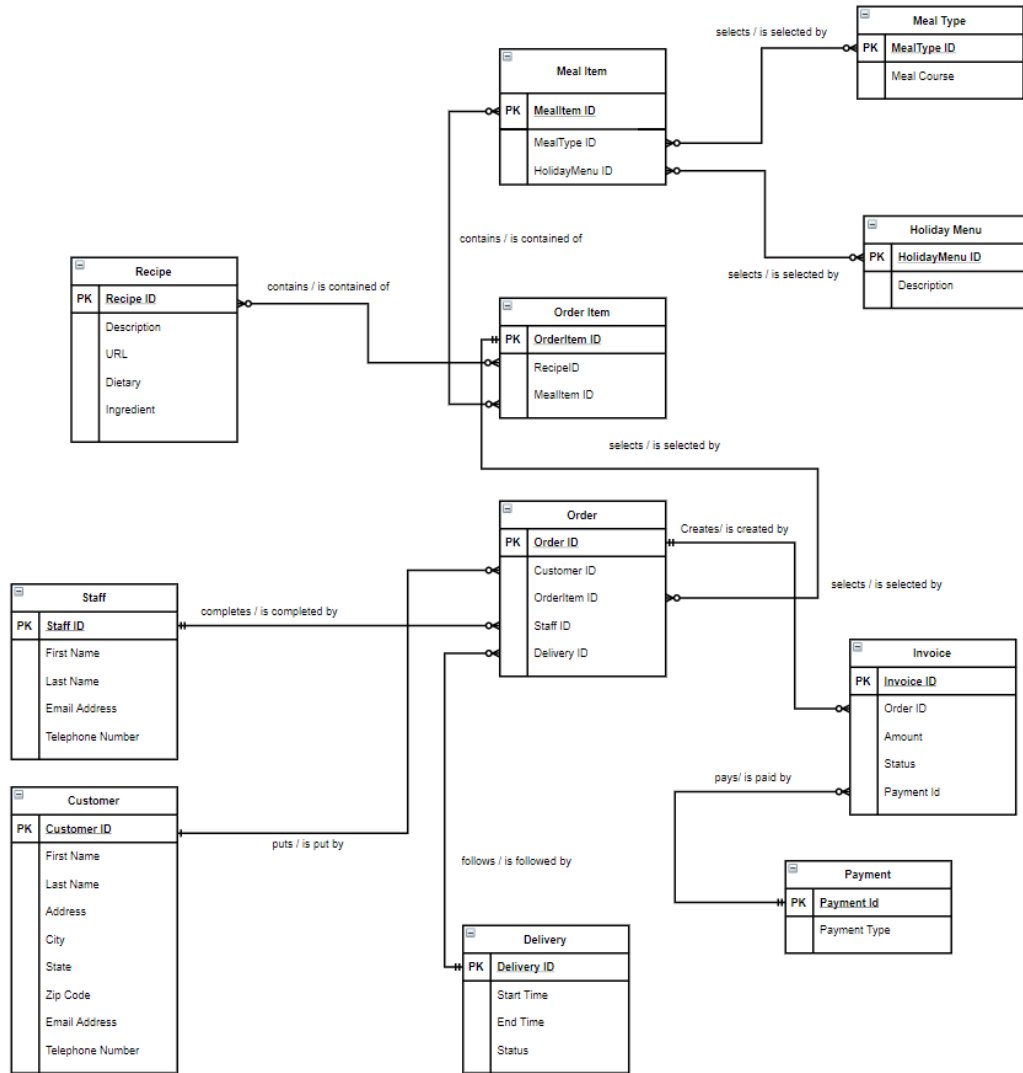
The **recipe** contains dietary preference, ingredient, recipe, quantity, and website on how the meal is prepared and recipe description.

Staff is an employee of Simple Kitchen, working in the meal prep or social media department.

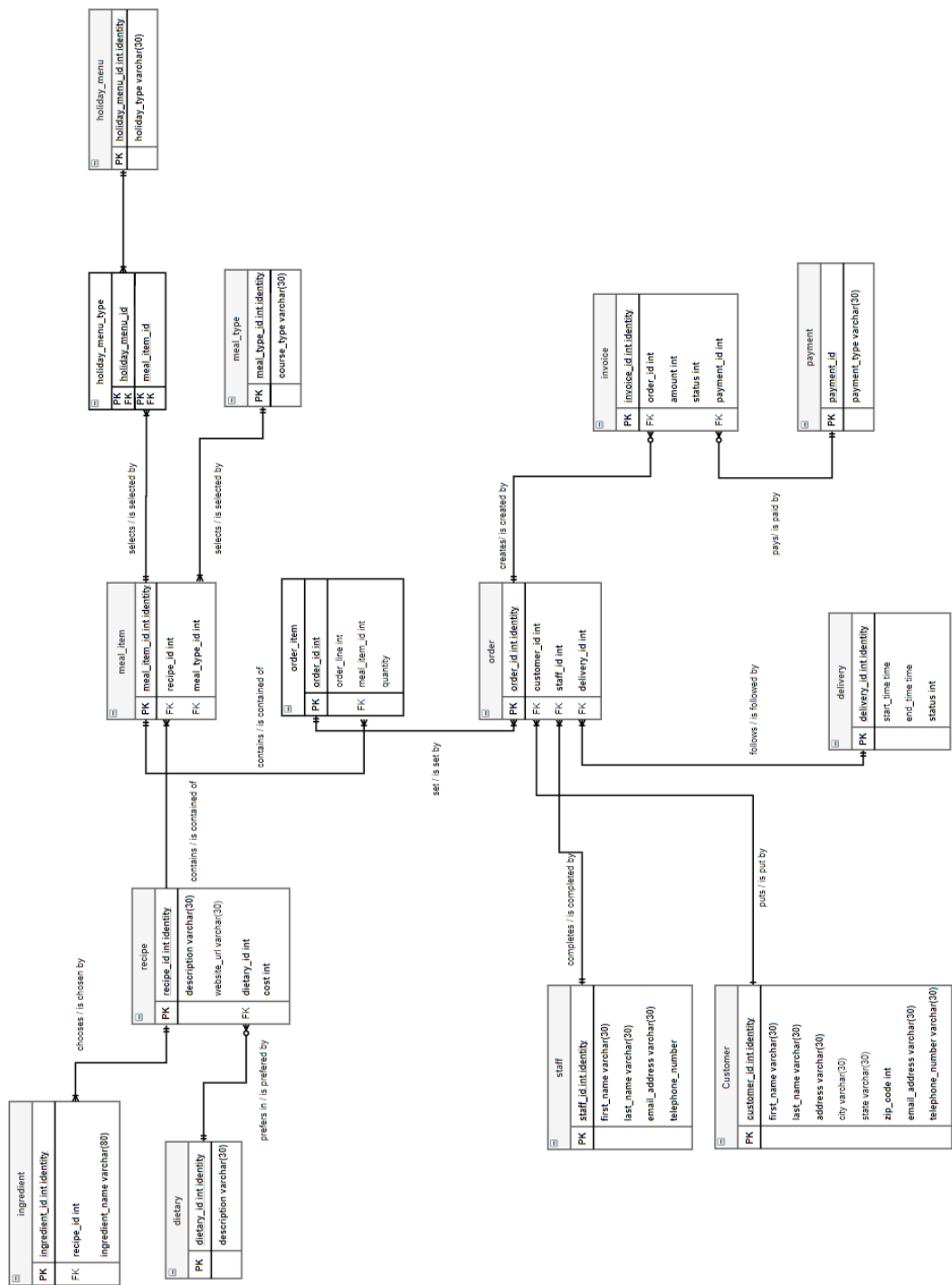
DATA QUESTIONS

1. What are the most serviced zip codes?
2. What is the most popular meal ordered?
3. How to identify a specific holiday dish ordered?
4. Which dietary choice is for increasing profitability?
5. Which customers have not paid their bills?

CONCEPTUAL MODEL



LOGICAL MODEL



PHYSICAL DATABASE DESIGN

--#1 dietary

```
CREATE TABLE dietary (  
    dietaryID int identity not null,  
    dietary_description varchar(50) not null,  
    CONSTRAINT PK_dietary PRIMARY KEY (dietaryID)  
)
```

--#2 recipes

```
CREATE TABLE recipes (  
    recipesID int identity not null,  
    website_url varchar(100) not null,  
    dietaryID int not null,  
    recipes_description varchar(100) not null,  
    cost int not null,  
    pricepoint int not null,  
    CONSTRAINT PK_recipes PRIMARY KEY (recipesID),  
    CONSTRAINT FK1_recipes FOREIGN KEY (dietaryID) REFERENCES dietary(dietaryID)  
)
```

--#3 ingredient

```
CREATE TABLE ingredient (  
    ingredientID int identity not null,  
    ingredient_description varchar(100) not null,  
    recipesID int not null,  
    CONSTRAINT PK_ingredient PRIMARY KEY (ingredientID),  
    CONSTRAINT FK1_ingredient FOREIGN KEY (recipesID) REFERENCES recipes(recipesID)  
)
```

--#4 meal_type

```
CREATE TABLE meal_type (  
    meal_typeID int identity not null,  
    meal_type_description varchar(100) not null,  
    CONSTRAINT PK_meal_type PRIMARY KEY (meal_typeID)
```



```

        meal_typeID int identity not null,
        course_type varchar(30) not null,
        CONSTRAINT PK_meal_type PRIMARY KEY (meal_typeID)
    )

--#5 holiday_menu
CREATE TABLE holiday_menu (
    holiday_menuID int identity not null,
    holiday_type varchar(30) not null,
    CONSTRAINT PK_holiday_menu PRIMARY KEY (holiday_menuID)
)

--#6 meal_item
CREATE TABLE meal_item (
    meal_itemID int identity not null,
    meal_typeID int not null,
    recipesID int not null,
    CONSTRAINT PK_meal_item PRIMARY KEY (meal_itemID),
    CONSTRAINT FK1_meal_item FOREIGN KEY (meal_typeID) REFERENCES
meal_type(meal_typeID),
    CONSTRAINT FK2_meal_item FOREIGN KEY (recipesID) REFERENCES recipes(recipesID)
)

--#7 holiday_menu_type
CREATE TABLE holiday_menu_type (
    holiday_menu_typeID int not null,
    holiday_menuID int not null,
    meal_itemID int not null,
    CONSTRAINT PK_holiday_menu_type PRIMARY KEY (holiday_menu_typeID),
    CONSTRAINT FK1_holiday_menu_type FOREIGN KEY (holiday_menu_typeID) REFERENCES
holiday_menu(holiday_menuID),

```

```
        CONSTRAINT FK2_holiday_menu_type FOREIGN KEY (meal_itemID) REFERENCES  
meal_item(meal_itemID)
```

```
)
```

```
--#8 staff
```

```
CREATE TABLE staff (
```

```
    staffID int identity not null,
```

```
    first_name varchar(30) not null,
```

```
    last_name varchar(30) not null,
```

```
    email_address varchar(30) not null,
```

```
    telephone_number varchar(30) not null,
```

```
    CONSTRAINT PK_staff PRIMARY KEY (staffID)
```

```
)
```

```
--#9 order_item
```

```
CREATE TABLE order_item (
```

```
    order_itemID int identity not null,
```

```
    meal_itemID int not null,
```

```
    order_line varchar(30) not null,
```

```
    quantity int not null
```

```
    CONSTRAINT PK_order_item PRIMARY KEY (order_itemID),
```

```
    CONSTRAINT FK1_order_item FOREIGN KEY (meal_itemID) REFERENCES  
meal_item(meal_itemID)
```

```
)
```

```
--#10 delivery
```

```
CREATE TABLE delivery (
```

```
    deliveryID int identity not null,
```

```
    start_time time not null,
```

```
    end_time time not null,
```

```
    delivery_status varchar(30) not null,
```

```
    delivery_date date not null,
```

```

        CONSTRAINT PK_delivery PRIMARY KEY (deliveryID)
    )

--#11 customer
CREATE TABLE customer (
    customerID int identity not null,
    first_name varchar(30) not null,
    last_name varchar(30) not null,
    customer_address varchar(30) not null,
    city varchar(30) not null,
    states varchar(30) not null,
    zip_code varchar(30) not null,
    email_addres varchar(30) not null,
    telephone_number varchar(30) not null,
    order_itemID int not null
    CONSTRAINT PK_customer PRIMARY KEY (customerID)
    CONSTRAINT FK1_customer FOREIGN KEY (order_item) REFERENCES order_item(order_itemID)
)

--#12 orders
CREATE TABLE orders (
    ordersID int identity not null,
    customerID int not null,
    order_itemID int not null,
    staffID int not null,
    deliveryID int not null,
    CONSTRAINT PK_orders PRIMARY KEY (ordersID),
    CONSTRAINT FK1_orders FOREIGN KEY (customerID) REFERENCES customer(customerID),
    CONSTRAINT FK2_orders FOREIGN KEY (order_itemID) REFERENCES order_item(order_itemID),
    CONSTRAINT FK3_orders FOREIGN KEY (staffID) REFERENCES staff(staffID),
    CONSTRAINT FK4_orders FOREIGN KEY (deliveryID) REFERENCES delivery(deliveryID)
)

```

)

--#13 payment

```
CREATE TABLE payment (  
    paymentID int identity not null,  
    payment_type varchar(30) not null,  
    CONSTRAINT PK_payment PRIMARY KEY (paymentID)
```

)

--#14 invoice

```
CREATE TABLE invoice (  
    invoiceID int identity not null,  
    ordersID int not null,  
    invoice_status varchar(30),  
    paymentID int not null,  
    CONSTRAINT PK_invoice PRIMARY KEY (invoiceID),  
    CONSTRAINT FK1_invoice FOREIGN KEY (ordersID) REFERENCES orders(ordersID),  
    CONSTRAINT FK2_invoice FOREIGN KEY (paymentID) REFERENCES payment(paymentID)
```

)

DATA CREATION

--#1 dietary

INSERT INTO dietary--add 7 diet choices

(dietary_description)

VALUES

('Regular'), ('Vegetarian'), ('Healthy'), ('Vegan'), ('Paleo'), ('Dairy-Free'), ('Gluten-Free')

Select * FROM dietary

--#2 recipes

INSERT INTO recipes--add 10 recipes

(website_url, dietaryID, recipes_description, cost, pricepoint)

VALUES

('https://www.simplyrecipes.com/recipes/slow_cooker_pumpkin_soup_vegan_paleo/', 2, 'Slow Cooker Pumpkin Soup!', \$4, \$10),

('https://https://www.simplyrecipes.com/recipes/honey_mustard_baked_salmon/', 3, 'Honey Mustard Salmon', \$10, \$25),

('https://www.simplyrecipes.com/recipes/moms_roast_turkey/', 1, 'Mom's Roast Turkey', \$40, \$90),

('https://www.simplyrecipes.com/recipes/prime_rib/', 5, 'Prime Rib', \$55, \$110),

('https://www.simplyrecipes.com/recipes/perfect_mashed_potatoes/', 2, 'Perfect Mashed Potatoes', \$2, \$10),

('https://www.simplyrecipes.com/recipes/beef_brisket/', 5, 'Beef Brisket', \$35, \$80),

('https://www.simplyrecipes.com/recipes/potato_latkes/', 2, 'Potato Latkes', \$2, \$15),

('https://www.simplyrecipes.com/recipes/spinach/', 4, 'Sautéed Spinach', \$4, \$10),

('https://www.simplyrecipes.com/recipes/crab_cakes/', 1, 'Dungeness Crab Cakes', \$10, \$30),

('https://www.simplyrecipes.com/recipes/shrimp_scampi/', 1, 'Shrimp Scampi', \$15, \$45)

Select * FROM recipes

--#3 ingredient

INSERT INTO ingredient--10 recipe ingredients

(ingredient_description, recipesID)

VALUES

('carrot', 6),
('onion', 6),
('celery', 6),
('ginger', 6),
('garlic', 6),
('bay leaves', 6),
('pumpkin', 6),
('coconut milk', 6),
('vegetable broth', 6),
('salt', 6),
('salmon fillets', 7),
('honey mustard', 7),
('garlic', 7),
('olive oil', 7),
('lemon juice', 7),
('dill', 7),
('turkey', 16),
('lemon juice', 16),
('salt', 16),
('pepper', 16),
('olive oil', 16),
('yellow onion', 16),
('celery', 16),
('parsley', 16),
('rosemary', 16),
('sage', 16),
('thyme', 16),
('rib roast', 17),

('salt', 17),
('black pepper', 17),
('yukon gold potatoes', 18),
('salt', 18),
('heavy cream', 18),
('butter', 18),
('milk', 18),
('salt', 18),
('black pepper', 18),
('beef brisket', 19),
('barbecue sauce', 19),
('soy sauce', 19),
('water', 19),
('russet potatoes', 20),
('onion', 20),
('all-purpose flour', 20),
('eggs', 20),
('salt', 20),
('black pepper', 20),
('canola oil', 20),
('sour cream', 20),
('spinach', 21),
('olive oil', 21),
('garlic', 21),
('salt', 21),
('crabmeat', 22),
('unsalted butter', 22),
('shallots', 22),
('kosher salt', 22),

('eggs', 22),
('worcestershire sauce', 22),
('sweet paprika', 22),
('black pepper', 22),
('tartar sauce', 22),
('lemon zest', 22),
('tabasco sauce', 22),
('parsley', 22),
('breadcrumbs', 22),
('shrimps', 23),
('olive oil', 23),
('butter', 23),
('salt', 23),
('garlic', 23),
('red pepper flakes', 23),
('white wine', 23),
('parsley', 23),
('black pepper', 23),
('lemon juice', 23)

SELECT * FROM ingredient

--#4 meal_type

INSERT INTO meal_type--add 2 meal courses

(course_type)

VALUES

('lunch'), ('dinner')

SELECT * FROM meal_type

--#5 holiday_menu


```
INSERT INTO holiday_menu--add 4 data
```

```
    (holiday_type)
```

```
VALUES
```

```
    ('Thanksgiving'), ('Hanukkah'), ('Christmas'), ('New Year')
```

```
Select * FROM holiday_menu
```

```
--#6
```

```
INSERT INTO meal_item
```

```
    (meal_typeID, recipesID)
```

```
VALUES
```

```
    (1, 1),
```

```
    (2, 1),
```

```
    (1, 2),
```

```
    (2, 2),
```

```
    (1, 3),
```

```
    (2, 3),
```

```
    (1, 4),
```

```
    (2, 4),
```

```
    (1, 5),
```

```
    (2, 5),
```

```
    (1, 6),
```

```
    (2, 6),
```

```
    (1, 7),
```

```
    (2, 7),
```

```
    (1, 8),
```

```
    (2, 8),
```

```
    (1, 9),
```

```
    (2, 9),
```

```
    (1, 10),
```

(2, 10)

SELECT * FROM meal_item

--#7 holiday_menu_type

INSERT INTO holiday_menu_type

(holiday_menuID, meal_itemID)

VALUES

(1, 1),

(1, 2),

(1, 5),

(1, 6),

(1, 9),

(1, 10),

(1, 15),

(1, 16),

(2, 3),

(2, 4),

(2, 9),

(2, 10),

(2, 11),

(2, 12),

(2, 13),

(2, 14),

(2, 15),

(2, 16),

(3, 1),

(3, 2),

(3, 3),

(3, 4),

(3, 5),

(3, 6),

(3, 7),

(3, 8),

(3, 9),

(3, 10),

(3, 11),

(3, 12),

(3, 13),

(3, 14),

(3, 15),

(3, 16),

(3, 17),

(3, 18),

(3, 19),

(3, 20),

(4, 1),

(4, 2),

(4, 3),

(4, 4),

(4, 5),

(4, 6),

(4, 7),

(4, 8),

(4, 9),

(4, 10),

(4, 11),

(4, 12),

(4, 13),

(4, 14),
(4, 15),
(4, 16),
(4, 17),
(4, 18),
(4, 19),
(4, 20)

SELECT * FROM holiday_menu_type

--#8 staff

INSERT INTO staff--add three Kitchen Table staffs

(first_name, last_name, email_address, telephone_number)

VALUES

('Maria', 'Ng', 'Mariang168@KitchenTable.com', 5103811869),
(('Herschel', 'Weinstock', 'HWeinstock@KitchenTable.com', 5107053445),
(('Nancy', 'Lee', 'NancyHein@KitchenTable.com', 5102228881)

Select * FROM staff

--#9 order_item

INSERT INTO order_item

(meal_itemID, order_line, quantity)

VALUES

(6, 'Moms Roast Turkey', 1),
(6, 'Moms Roast Turkey', 2),
(10, 'Perfect Mashed Potatoes', 1),
(14, 'Sauteed Spinach', 1),
(6, 'Moms Roast Turkey', 1),
(12, 'Beef Brisket', 2),
(6, 'Moms Roast Turkey', 1),

```
(4, 'Honey Mustard Salmon', 2),  
(8, 'Prime Rib', 1),  
(20, 'Shrimp Scampi', 2)
```

```
SELECT * FROM order_item
```

```
--#10 delivery
```

```
INSERT INTO delivery--add 10 delivery info
```

```
(start_time, end_time, delivery_status, delivery_date)
```

```
VALUES
```

```
('11:00 am', '12:00 pm','Completed','11/23/2020'),  
( '5:00 pm', '5:20 pm','Completed','11/23/2020'),  
( '2:00 pm', '3:00 pm','Completed','11/25/2020'),  
( '11:30 am', '12:00 pm','Completed','11/25/2020'),  
( '11:00 am', '12:30 pm','Completed','11/25/2020'),  
( '5:00 pm', '6:00 pm','Scheduled','12/10/2020'),  
( '6:00 pm', '6:30 pm','Completed','11/25/2020'),  
( '5:00 pm', '6:00 pm','In Progress','12/05/2020'),  
( '6:00 pm', '6:45 pm','Scheduled','12/23/2020'),  
( '5:30 pm', '6:15 pm','Scheduled','12/09/2020')
```

```
SELECT * FROM delivery
```

```
--#11
```

```
INSERT INTO customer
```

```
(first_name, last_name, customer_address, city, states, zip_code, email_address,  
telephone_number, order_itemID)
```

```
VALUES
```

```
('Boxer', 'Kevin', '661 Logan Crossing', 'Hayward', 'CA', 94544, 'kboxer0@free.fr', 2099651776,  
1),
```

```
('Kliment', 'L Estrange', '124 Toban Park', 'San Francisco', 'CA', 94105, 'klestrange1@e-  
recht24.de', 6509241659, 2),
```

('Eduard', 'Goodman', '33 Eastlawn Way', 'San Jose', 'CA', 95173, 'egoodman2@google.cn', 4083042819, 3),

('Austen', 'Bascombe', '49211 Acker Alley', 'Berkeley', 'CA', 94805, 'abascombe3@bandcamp.com', 9098281609, 4),

('Vinni', 'Leisk', '903 Blaine Alley', 'Oakland', 'CA', 94605, 'vleisk4@amazon.com', 5102277326, 5),

('Viole', 'Shattock', '09 Calypso Place', 'Berkeley', 'CA', 94805, 'vshattock5@1und1.de', 3102030154, 6),

('Anthia', 'Hern', '80804 Springview Parkway', 'San Francisco', 'CA', 94105, 'aohern6@sfgate.com', 2095554229, 7),

('Lisbeth', 'Retter', '3 Center Street', 'Berkeley', 'CA', 94805, 'lretter7@blogspot.com', 9167584453, 8),

('Tildy', 'Worswick', '6484 Mallard Point', 'San Francisco', 'CA', 94105, 'tworswick8@sina.com.cn', 4157264794, 9),

('Flossie', 'Dunniom', '8 Debra Circle', 'Berkeley', 'CA', 94805, 'fodunniom9@sun.com', 4086657387, 10)

SELECT * FROM customer

--#12 orders

INSERT INTO orders--10 orders

(customerID, order_itemID, staffID, deliveryID)

VALUES

(1, 1, 1, 1),

(2, 2, 2, 2),

(3, 3, 2, 3),

(4, 4, 1, 4),

(5, 5, 1, 5),

(6, 6, 3, 6),

(7, 7, 3, 7),

(8, 8, 2, 8),

(9, 9, 2, 9),

(10, 10, 2, 10)

```
SELECT * FROM orders
```

```
--#13
```

```
INSERT INTO payment--add 3 options
```

```
    (payment_type)
```

```
VALUES
```

```
    ('Venmo'), ('Zelle'), ('Paypal')
```

```
Select * FROM payment
```

```
--#14
```

```
INSERT INTO invoice
```

```
    (ordersID, invoice_status, paymentID)
```

```
VALUES
```

```
    (1, 'paid', 1),
```

```
    (2, 'paid', 1),
```

```
    (3, 'paid', 1),
```

```
    (4, 'paid', 2),
```

```
    (5, 'paid', 1),
```

```
    (6, 'paid', 3),
```

```
    (7, 'paid', 2),
```

```
    (8, 'pending', 2),
```

```
    (9, 'pending', 2),
```

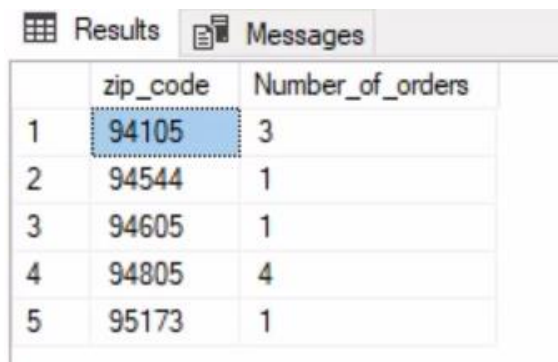
```
    (10, 'pending', 2)
```

```
SELECT * FROM invoice
```

DATA MANIPULATION AND ANSWERING DATA QUESTION

1. What are the most serviced zip codes?

```
SELECT
    customer.zip_code,
    count(orders.ordersID) AS Number_of_orders
FROM customer
JOIN orders ON customer.customerID=orders.customerID
GROUP by customer.zip_code
Order by customer.zip_code
```



The screenshot shows a database interface with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with the following data:

	zip_code	Number_of_orders
1	94105	3
2	94544	1
3	94605	1
4	94805	4
5	95173	1

Figure 1: Business Question 1

Answer 1: Zip code 94805 has the most orders.

2. What is the most popular meal ordered?

```
SELECT
    recipes.recipes_description,
    sum(order_item.quantity) as meals_ordered
FROM order_item
JOIN meal_item on order_item.meal_itemID=meal_item.meal_itemID
JOIN recipes on recipes.recipesID=meal_item.recipesID
GROUP by recipes.recipes_description
Order by meals_ordered desc
```


	recipes_description	meals_ordered
1	Mom's Roast Turkey	5
2	Beef Brisket	2
3	Honey Mustard Salmon	2
4	Shrimp Scampi	2
5	Perfect Mashed Potatoes	1
6	Potato Latkes	1
7	Prime Rib	1

Figure 2: Business Question 2

Answer 3: Identified Moms Roast Turkey has 5 meals ordered.

3. How to identify a specific holiday dish ordered?

```
SELECT * FROM order_item
```

```
WHERE order_line='Moms Roast Turkey'
```

	order_itemID	meal_itemID	order_line	quantity
1	1	6	Moms Roast Turkey	1
2	2	6	Moms Roast Turkey	2
3	5	6	Moms Roast Turkey	1
4	7	6	Moms Roast Turkey	1

Figure 3: Business Question 3

Answer 3: Entered Moms Roast Turkey or other recipes

4. Which dietary choice is for increasing profitability?

```
SELECT
```

```
    dietary.dietary_description,
```

```
    recipes.recipes_description,
```

```
    (recipes.pricepoint-recipes.cost) as Profitability_per_serving
```

```
FROM recipes
```

```
JOIN dietary on recipes.dietaryID=dietary.dietaryID
```

```
order by dietary.dietary_description, recipes.recipes_description
```

Results		Messages	
	dietary_description	recipes_description	Profitability_per_serving
1	Healthy	Honey Mustard Salmon	15
2	Paleo	Beef Brisket	45
3	Paleo	Prime Rib	55
4	Regular	Dungeness Crab Cakes	20
5	Regular	Mom's Roast Turkey	50
6	Regular	Shrimp Scampi	30
7	Vegan	Sautéed Spinach	6
8	Vegetarian	Perfect Mashed Potatoes	8
9	Vegetarian	Potato Latkes	13
10	Vegetarian	Slow Cooker Pumpkin Soup!	6

Figure 4: Business Question 4

Answer 4: Paleo dietary seemed to be highly profitable for Kitchen Table.

5. Which customers have not paid their bills?

CREATE VIEW Payment_pending

AS

SELECT

customer.first_name,

customer.last_name,

invoice.invoice_status,

recipes.recipes_description,

order_item.quantity as quantity,

(recipes.pricepoint*order_item.quantity) as customer_billing

FROM invoice

Join orders on invoice.ordersID = orders.ordersID

Join order_item on orders.order_itemID = order_item.order_itemID

JOIN meal_item on order_item.meal_itemID = meal_item.meal_itemID

JOIN recipes on meal_item.recipesID = recipes.recipesID

Join customer on orders.customerID = customer.customerID

WHERE invoice.invoice_status = 'pending'

```
SELECT * FROM Payment_pending
```

Results Messages						
	first_name	last_name	invoice_status	recipes_description	quantity	customer_billing
1	Lisbeth	Retter	pending	Honey Mustard Salmon	2	50
2	Tildy	Worswick	pending	Prime Rib	1	110
3	Flossie	Dunniom	pending	Shrimp Scampi	2	90

Figure 5: Business Question 5

Answer 5: Lisbeth Retter, Tildy Worswick and Flossie Dunniom owed Kitchen Table from their recent orders.

IMPLEMENTATION

```
require(RODBC)
myconn<-odbcConnect("VidCast64")
sqlSelectStatement<-
"SELECT
    customer.zip_code,
    count(orders.ordersID) AS Number_of_orders
FROM customer
JOIN orders ON customer.customerID=orders.customerID
GROUP by customer.zip_code
Order by customer.zip_code
"

sqlResult <-sqlQuery(myconn, sqlSelectStatement)

library(ggplot2)

df<-data.frame(zip_code=c("94105","94544","94605","94805","95173"),
Number_of_orders=c(3,1,1,4,1))

plot1<-ggplot(data=df, aes(x=zip_code, y=Number_of_orders)) + geom_bar(stat="identity",
fill="light blue")

plot1
```

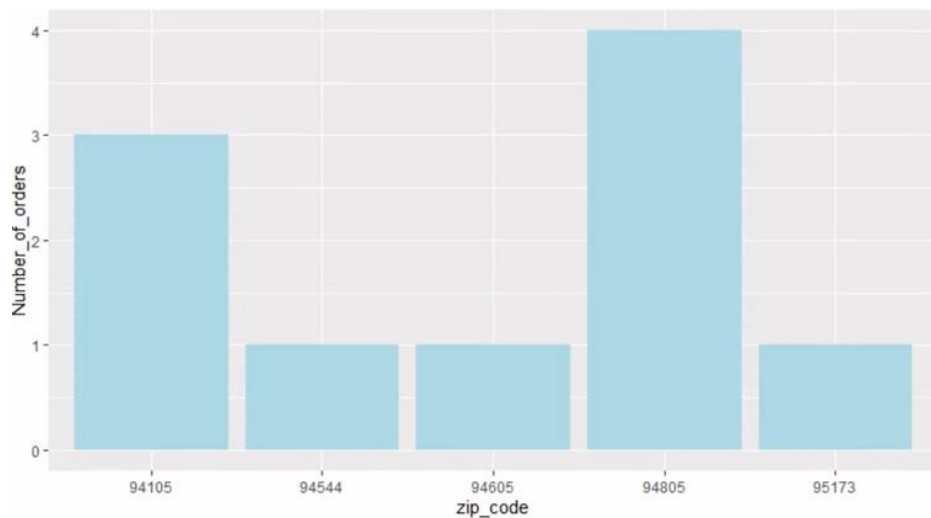


Figure 6: Barplot of Business Question 1-What are the most serviced zip codes?

```

sqlSelectStatement1<-
  "SELECT
    recipes.recipes_description,
    sum(order_item.quantity) as meals_ordered
  FROM order_item
  JOIN meal_item on order_item.meal_itemID=meal_item.meal_itemID
  JOIN recipes on recipes.recipesID=meal_item.recipesID
  GROUP by recipes.recipes_description
  Order by meals_ordered desc
  "

sqlResult1 <-sqlQuery(myconn, sqlSelectStatement1)

df1<-data.frame(recipes_description=c("Mom's Roast Turkey","Beef Brisket","Honey Mustard
Salmon","Shrimp Scampi","Perfect Mashed Potatoes","Potato Latkes","Prime Rib"),
meals_ordered=c(5,2,2,2,1,1,1))

plot2<-ggplot(data=df1, aes(x=recipes_description, y=meals_ordered)) +
geom_bar(stat="identity", fill="light green")

plot2 + scale_x_discrete(limits=c("Mom's Roast Turkey","Beef Brisket","Honey Mustard
Salmon","Shrimp Scampi","Perfect Mashed Potatoes","Potato Latkes","Prime Rib"))

```

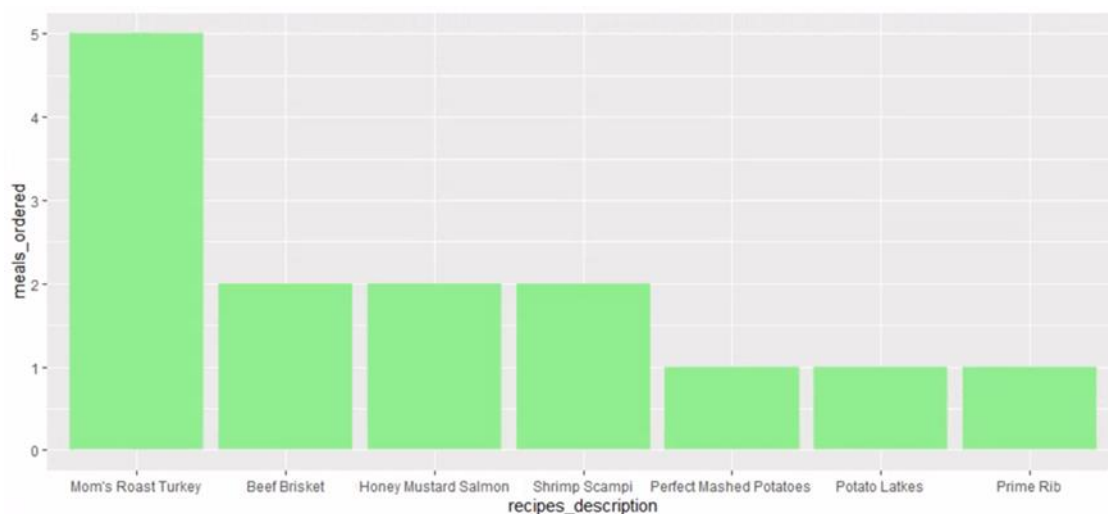


Figure 7: Barplot of Business Question 2 - What is the most popular meal ordered?

```
sqlSelectStatement2<-
```

```
"SELECT
```

```
    customer.first_name,
```

```
    customer.last_name,
```

```
    invoice.invoice_status,
```

```
    recipes.recipes_description,
```

```
    order_item.quantity as quantity,
```

```
    (recipes.pricepoint*order_item.quantity) as customer_billing
```

```
FROM invoice
```

```
Join orders on invoice.ordersID = orders.ordersID
```

```
Join order_item on orders.order_itemID = order_item.order_itemID
```

```
JOIN meal_item on order_item.meal_itemID = meal_item.meal_itemID
```

```
JOIN recipes on meal_item.recipesID = recipes.recipesID
```

```
Join customer on orders.customerID = customer.customerID
```

```
WHERE invoice.invoice_status = 'pending'
```

```
"
```

```
sqlResult2 <-sqlQuery(myconn, sqlSelectStatement2)
```

```
df2<-data.frame(customer=c("Lisbeth Retter", "Tildy Worswick", "Flossie Dunnium"),
```

```
    pending_bills=c(50, 110, 90))
```

```
plot3<-ggplot(data=df2, aes(x=customer, y=pending_bills)) + geom_bar(stat="identity",  
fill="pink")
```

```
plot3 + scale_x_discrete(limits=c("Lisbeth Retter", "Tildy Worswick", "Flossie Dunnium"))
```

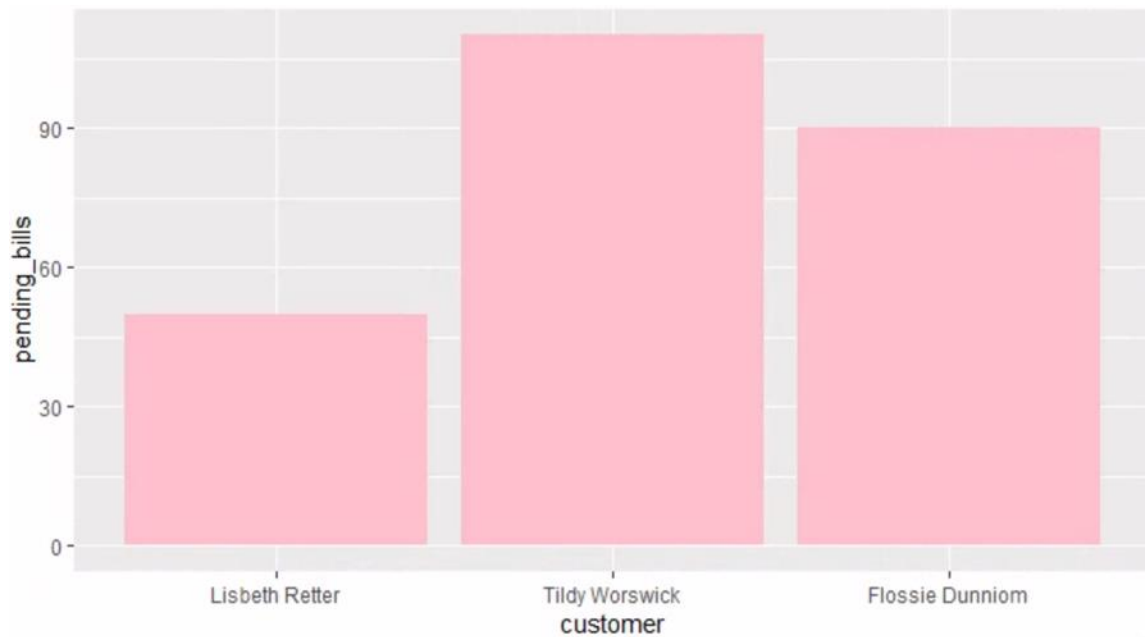


Figure 8: Barplot Business Question 5 - Which customers have not paid their bills?

odbcCloseAldf2

REFLECTION

This project is my first attempt at creating a database and learning to appreciate the pain. I am glad to draft a conceptual and logical model before building a physical model in the SQL environment. Then, finding minor connections that will not work well for my business questions and found myself deleting my database more than four times. I think the final version of my database for this project is version six. Upon reflection, every time I deleted my database, I increased my efficiency and speed in specific SQL codings. Also, I appreciate the database developer having the patience and perseverance to work through more than hundreds of tables and unlimited data.

I have to admit that my fourteen tables and ten rows of data have given me pains and frustrations, but I did learn a lot. The lab works were very helpful in guiding me through my project. I played catch up some of the times, and I wished to spend more time working to correctly create a view or procedure. But I managed to perform the correct view coding finally for my project. I did enjoy the process of completing my SQL database and able to pull my data into RStudio for data visualization. Lastly, I wish that I can retake this class and build a better database!

Now I understand why professor Chad emphasized following the labwork very closely.

SUMMARY

Understanding what business questions you seek to answer are the crucial foundation of your conceptual and logical models. Once one understands the problems, the drafting out of the conceptual model will be easier. However, the whole process has to go through a few iterations before starting a physical model. There is no one perfect physical model that one can use forever. The entire process must allow the database developer to perform continuous improvement. I found that SQL and Rstudio's usage provides me an easy way to provide critical insights into my Kitchen Table clients for them on how to improve their business. Also, I noticed that Microsoft Access usage also provides ease of operation on creating relationships and forms. I found many things to improve on my database, but I genuinely appreciate the project's completion. Also, motivate me to fine-tune my database coding skillset and create a better database.