California Polytechnic State University,
San Luis Obispo

Data Management for Business Analytics
GSB S520

POPUP BUSINESS MODEL

Author:
Andy Chiv, Joseph Lee, Kriti Saxena, Kyle Pelham

Professor:
Yongcheng Zang

November 15, 2022

Table of Contents

1. **Business Scope and Process**

SLO Popup is a business model with the aim to connect local residents and Cal Poly students to local popups in the San Luis Obispo area. Each popup can sell food items on the SLO Popup website and customers will find it convenient to taste different food options run by local popups. Our goal is to provide multiple highly-rated food options to people who may not have access to the variety of cuisines that are available in the SLO area. In doing so, we are providing more business opportunities for San Luis Obispo commerce and providing much needed exposure to small business owners. By creating a database to handle the transactions, we are providing the necessary means to thrive in SLO as a restaurant or food business owner as well as connecting the SLO population with food they otherwise would never have known about.

Customers will have many options to order from various popup stores online and also be able to choose from various payment options such as Google Pay, Apple Pay, PayPal, and credit card. They will also be given the choices to order multiple food listings with different quantities in one order from the same store. Multiple delivery modes will be available including Doordash and Uber, and a customer pick-up option will be provided.

Ordering processes are as follows:

1. ***Customer Ordering:*** Customers will order food from different popups online by providing their name, email address, and phone number.
2. ***Choosing delivery services:*** Customers will select a delivery service (Uber or Doordash, or self pickup) when checking out. If a customer selects self pickup, they will be given their approximate pickup time. Otherwise, they will be given delivery service information so the customer can contact the driver if necessary.
3. ***Selecting payment options:*** Payment options are available such as Google Pay, Apple Pay, PayPal, and card, all of which have their own unique identification. Payment ID and customer ID will be recorded so that we know which payment belongs to which customers.
4. ***Processing the orders:*** Popup receives the order requests from the customers and each popup has access to each order information, customer information, delivery chosen by customers, payment information and the order date.

2. Business Rules

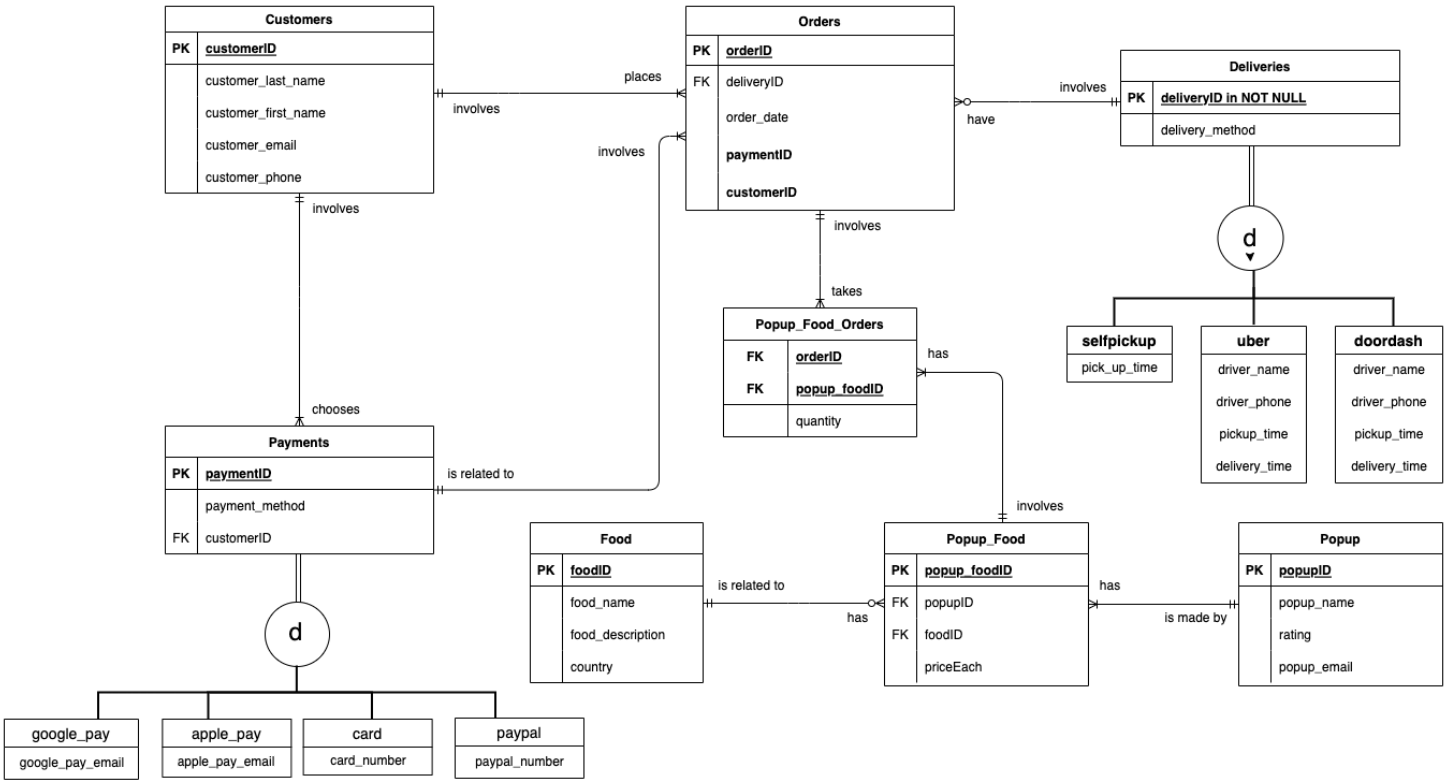
1. Each customer can have many payments and there are 4 payment methods: google pay, apple pay, card, paypal. Each payment is only related to one and only one customer.
2. Each payment can have one or many orders. Each order can have one and only one payments
3. Each customer can have one or many orders. Each order can have only one customer.
4. Each order can be delivered by only one delivery service and each delivery service can take zero or many orders. There are three delivery methods including self-pickup, Uber and doordash.
5. Each order can have many popup foods. Each popup food can be placed by many orders.
6. Each popup can have many foods and each food can exist in many popups.

3. Definition

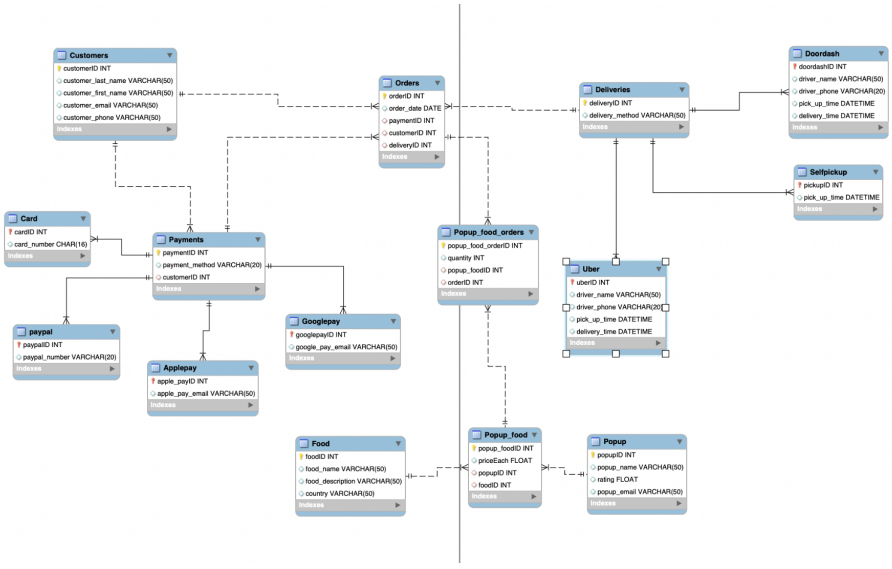
1. **Customers: someone who place orders of food from popup stores**
    a. customerID
    b. customer_last_name
    c. customer_first_name
    d. customer_email
    e. customer_ phone
2. **Orders: orders placed by customers**
    a. orderID
    b. deliveryID
    c. order_date
    d. paymentID
    e. customerID
3. **Payments: stores the available customer payment methods. There are only four payment methods**
    a. paymentID
    b. payment_method
    c. customerID
4. **google_pay: first payment method**
    a. google_pay_email
5. **apple_pay: second payment method**
    a. apple_pay_email
6. **card: third payment method**
    a. card_number
7. **paypal: fourth payment method**
    a. paypal_number

8. **Food: food that are available in popup which can be placed in an order**
    a. foodID
    b. food_name
    c. food_description
    d. country
9. **Popup: store that sells food**
    a. popupID
    b. popup _name
    c. rating
    d. popup_email
10. **Popup_food: connect food and popup to determine price for each food listing**
    a. popup_foodID
    b. popupID
    c. foodID
    d. priceEach
11. **Popup_Food_Orders: connect each food with price to determine the quantity ordered**
    a. orderID
    b. popup_foodID
    c. quantity
12. **Deliveries: delivery services a customer can choose. There are three available options.**
    a. deliveryID
    b. delivery_method
13. **selfpickup: first delivery method**
    a. pick_up_time
14. **uber: second delivery method**
    a. driver_name
    b. driver_phone
    c. pickup_time
    d. delivery_time
15. **doordash: third delivery method**
    a. driver_name
    b. driver_phone
    c. pickup_time
    d. delivery_time

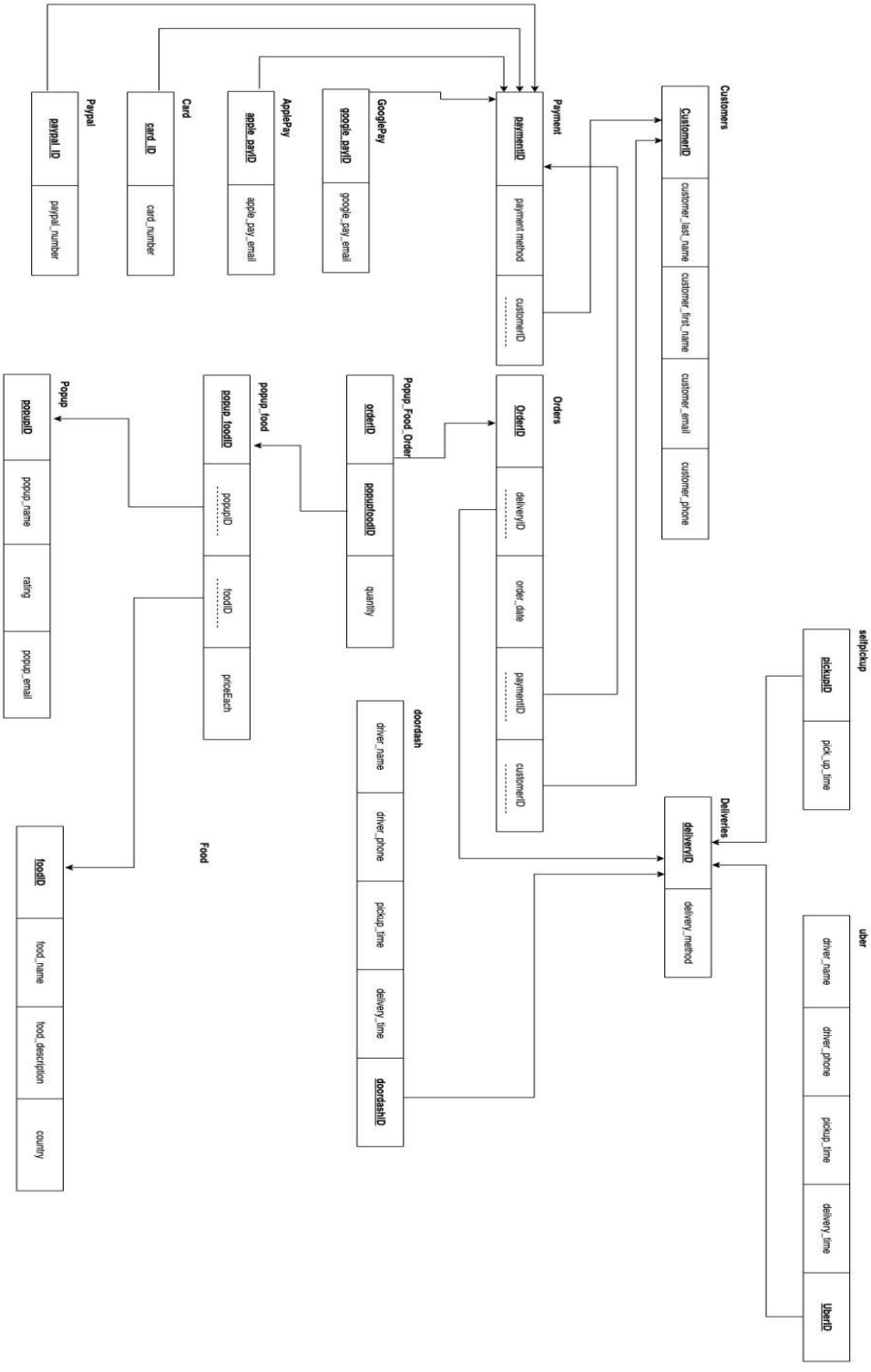
4. Enhanced Entity Relationship (EER) Diagram:



Since the database was designed from scratch, normalization concepts were already taken into account in the logical design. As a result, the normalization step is not needed because every table was created with third-normal form in mind. Therefore, we did not have to perform any further normalization on any of the tables in our database design.



5. Logical Design:



6. Create Databases

```
CREATE DATABASE IF NOT EXISTS popup;
USE popup;
```

```
CREATE TABLE IF NOT EXISTS Customers (
customerID INT,
customer_last_name VARCHAR(50),
customer_first_name VARCHAR(50),
customer_email VARCHAR(50),
customer_phone VARCHAR(50),
PRIMARY KEY(customerID)
);
```

```
CREATE TABLE IF NOT EXISTS Payments (
paymentID INT,
payment_method varchar(20),
customerID INT,
PRIMARY KEY (paymentID),
FOREIGN KEY (customerID) REFERENCES Customers (customerID)
);
```

```
CREATE TABLE IF NOT EXISTS Googlepay (
googlepayID INT,
google_pay_email VARCHAR(50),
PRIMARY KEY (googlepayID),
FOREIGN KEY (googlepayID) REFERENCES Payments (paymentID)
);
```

```
CREATE TABLE IF NOT EXISTS Applepay (
apple_payID INT,
apple_pay_email VARCHAR(50),
PRIMARY KEY (apple_payID),
FOREIGN KEY (apple_payID) REFERENCES Payments (paymentID)
);
```

```
CREATE TABLE IF NOT EXISTS Card (
cardID INT,
card_number CHAR(16),
PRIMARY KEY (cardID),
FOREIGN KEY (cardID) REFERENCES Payments (paymentID));
```

```sql
CREATE TABLE IF NOT EXISTS paypal (
paypalID INT,
paypal_number VARCHAR(20),
PRIMARY KEY (paypalID),
FOREIGN KEY (paypalID) REFERENCES Payments (paymentID)
);
```

```sql
CREATE TABLE IF NOT EXISTS Deliveries (
deliveryID INT,
delivery_method VARCHAR(50),
PRIMARY KEY(deliveryID)
);
```

```sql
CREATE TABLE IF NOT EXISTS Selfpickup (
pickupID INT,
pick_up_time DATETIME,
PRIMARY KEY (pickupID),
FOREIGN KEY (pickupID) REFERENCES deliveries (deliveryID)
);
```

```sql
CREATE TABLE IF NOT EXISTS Uber (
uberID INT,
driver_name VARCHAR(50),
driver_phone VARCHAR(20),
pick_up_time DATETIME,
delivery_time DATETIME,
PRIMARY KEY (uberID),
FOREIGN KEY (uberID) REFERENCES deliveries (deliveryID)
);
```

```sql
CREATE TABLE IF NOT EXISTS Doordash (
doordashID INT,
driver_name VARCHAR(50),
driver_phone VARCHAR(20),
pick_up_time DATETIME,
delivery_time DATETIME,
PRIMARY KEY (doordashID),
FOREIGN KEY (doordashID) REFERENCES deliveries (deliveryID)
);
```

```
CREATE TABLE IF NOT EXISTS Orders (
orderID INT,
order_date DATE,
paymentID INT,
customerID INT,
deliveryID INT,
PRIMARY KEY (orderID),
FOREIGN KEY (customerID) REFERENCES Customers (customerID),
FOREIGN KEY (deliveryID) REFERENCES Deliveries (deliveryID),
FOREIGN KEY (paymentID) REFERENCES Payments (paymentID)
);
```

```
CREATE TABLE IF NOT EXISTS Food (
foodID INT,
food_name VARCHAR(50),
food_description VARCHAR(50),
country VARCHAR(50),
PRIMARY KEY(foodID)
);
```

```
CREATE TABLE IF NOT EXISTS Popup (
popupID INT,
popup_name VARCHAR(50),
rating FLOAT,
popup_email VARCHAR(50),
PRIMARY KEY(popupID)
);
```

```
CREATE TABLE IF NOT EXISTS Popup_food (
popup_foodID INT,
priceEach FLOAT,
popupID INT,
foodID INT,
PRIMARY KEY(popup_foodID),
FOREIGN KEY (popupID) REFERENCES Popup (popupID),
FOREIGN KEY (foodID) REFERENCES Food (foodID)
);
```

```
CREATE TABLE IF NOT EXISTS Popup_food_orders (
popup_food_orderID INT,
quantity INT,
popup_foodID INT,
orderID INT,
PRIMARY KEY(popup_food_orderID),
FOREIGN KEY (popup_foodID) REFERENCES Popup_food (popup_foodID),
FOREIGN KEY (orderID) REFERENCES orders (orderID)
);
```

7. Populate the Database

```
INSERT INTO Customers (customerID, customer_last_name, customer_first_name , customer_email ,
customer_phone ) VALUES ('1', 'saxena', 'kriti', 'kesaxena@gmail.com', '805-805-1357');
INSERT INTO Customers (customerID, customer_last_name, customer_first_name , customer_email ,
customer_phone ) VALUES ('2', 'lee', 'joseph', 'jlee932cp@gmail.com', '213-276-2724');
INSERT INTO Customers (customerID, customer_last_name, customer_first_name , customer_email ,
customer_phone ) VALUES ('3', 'pelham', 'kyle ', 'kyp123@gmail.com', '699-123-4567');
INSERT INTO Customers (customerID, customer_last_name, customer_first_name , customer_email ,
customer_phone ) VALUES ('4', 'chiv', 'andy', 'andy204@gmail.com', '805-750-3461');
INSERT INTO Customers (customerID, customer_last_name, customer_first_name , customer_email ,
customer_phone ) VALUES ('5', 'zhan', 'yong', 'zyong@hotmail.com ', '805-124-5478');
```

```
INSERT INTO Payments (paymentID , payment_method, customerID) VALUES ( '101', 'apple_pay', '1');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('102', 'google_pay', '2');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('103', 'card', '3');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('104', 'card', '4');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('105', 'paypal', '5');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('106', 'paypal', '4');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('107', 'google_pay', '4');
INSERT INTO Payments (paymentID , payment_method , customerID) VALUES ('108', 'apple_pay', '2');
```

```
INSERT INTO googlepay (googlepayID, google_pay_email) VALUES ('101', 'jlee932cp@gmail.com');
INSERT INTO googlepay (googlepayID, google_pay_email) VALUES ('107', 'andy204@gmail.com');
```

```
INSERT INTO Applepay (apple_payID, apple_pay_email) VALUES ('101', 'kesaxena@gmail.com');
INSERT INTO Applepay (apple_payID, apple_pay_email) VALUES ('108', 'jlee932cp@gmail.com');
```

```
insert into card (cardID, card_number) values ('103', '4426888877776666');
insert into card (cardID, card_number) values ('104', '4111222233334444');
```

```
insert into paypal (paypalID, paypal_number) values ('105', '805-124-5478');
insert into paypal (paypalID, paypal_number) values ('106', '805-750-3461');
```

```
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1001', 'selfpickup');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1002', 'uber');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1003', 'doordash');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1004', 'selfpickup');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1005', 'doordash');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1006', 'doordash');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1007', 'uber');
INSERT INTO Deliveries (deliveryID, delivery_method ) VALUES ('1008', 'doordash');
```

```
INSERT INTO Doordash (doordashID, driver_name, driver_phone, pick_up_time, delivery_time) VALUES
('1003', 'Daniel Ric', '3057041345', '2022-04-24 11:00:00', '2022-04-24 11:20:00');
INSERT INTO Doordash (doordashID, driver_name, driver_phone, pick_up_time, delivery_time) VALUES
('1005', 'Karl Max ', '6993051257', '2020-03-10 18:00:00', '2020-03-10 18:25:00');
INSERT INTO Doordash (doordashID, driver_name, driver_phone, pick_up_time, delivery_time) VALUES
('1006', 'Vance Miller', '8468885555', '2020-03-13 15:00:00', '2020-03-13 15:20:00');
INSERT INTO Doordash (doordashID, driver_name, driver_phone, pick_up_time, delivery_time) VALUES
('1008', 'Lewis Hamilton', '8468345555', '2020-10-10 19:00:00', '2020-10-10 20:00:00');
```

```
INSERT INTO Uber (uberID, driver_name, driver_phone, pick_up_time, delivery_time) VALUES ('1002', 'John
Smith', '8052224444', '2021-11-20 16:15:00', '2021-11-20 16:30:00');
INSERT INTO Uber (uberID, driver_name, driver_phone, pick_up_time, delivery_time) VALUES ('1007', 'Larry
Elder', '8054567890', '2021-04-24 12:30:00', '2021-04-24 12:45:00');
```

```
INSERT INTO Selfpickup (pickupID, pick_up_time ) VALUES ('1001', '2022-01-02 12:15:00');
INSERT INTO Selfpickup (pickupID, pick_up_time ) VALUES ('1004', '2022-03-13 11:00:00');
```

```
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('101', '2021-11-20',
'101', '1', '1001');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('102', '2022-01-02',
'102', '2', '1002');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('103', '2022-04-02',
'103', '3', '1003');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('104', '2021-04-24',
'104', '4', '1004');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('105', '2020-03-10',
'105', '5', '1005');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('106', '2020-03-13',
'106', '4', '1006');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('107', '2022-03-13',
'107', '4', '1007');
INSERT INTO Orders (orderID, order_date, paymentID, customerID, deliveryID) VALUES ('108', '2020-10-10',
'108', '2', '1008');
```

```
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('11', 'Floating bar', '4.8',
'customercare@fbar.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('22', 'Frank\'s Frankfurters', '5',
'customercare@ff.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('33', 'Thai Palace', '4.5',
'customercare@tp.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('44', 'Spaghetti Factory', '5',
'customercare@sf.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('55', 'Mel\'s Diner', '3.5',
'customercare@md.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('66', 'Fredy\'s BBQ ', '3',
'customercarefbbq@gmail.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('77', 'Papa\'s kitchen', '4.2',
'customercare@pk.com');
INSERT INTO Popup (popupID, popup_name, rating, popup_email) VALUES ('88', 'Dhaba 65', '5',
'customercareD65@gmail.com');
```

```
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('111', 'tacos', 'with beef and
extra onions ', 'Mexico');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('222', 'hotdog', 'hot dog
without mustard ', 'US');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('333', 'pad thai', 'chicken pad
thai with green onions', 'Thailand');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('444', 'spaghetti', 'spaghetti
and meatballs', 'Italy');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('555', 'hamburger', 'double
burger with cheese', 'US');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('666', 'southern style bbq ribs
', 'extra sauce ', 'US');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('777', 'Korean fried chicken',
'Extra spicy ', 'Korean');
INSERT INTO Food (foodID, food_name, food_description, country) VALUES ('888', 'Chicken tikka masala',
'with garlic butter naan ', 'India');
```

```
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('1', '10', '11', '111');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('2', '15', '11', '222');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('3', '12', '22', '444');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('4', '11', '33', '555');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('5', '16', '33', '666');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('6', '10', '33', '777');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('7', '9', '44', '888');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('8', '15', '55', '111');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('9', '13', '66', '111');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('10', '13', '77', '444');
INSERT INTO Popup_food (popup_foodID , priceEach, popupID, foodID) VALUES ('11', '10', '88', '333');
```

```
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('1', '3',
'1', '101');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('2', '2',
'2', '101');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('3', '2',
'4', '102');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('4', '1',
'5', '102');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('5', '5',
'6', '102');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('6', '5',
'3', '103');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('7', '4',
'7', '104');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('8', '2',
'8', '105');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('9', '8',
'9', '106');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('10', '9',
'10', '107');
INSERT INTO Popup_food_orders (popup_food_orderID, quantity, popup_foodID , orderID) VALUES ('11', '5',
'11', '108');
```

Display all database information using SQL Statements

```
150 •  select*from Customers;
151
```

100% 4:150

Result Grid Filter Rows: Search Edit: Export/Import:

customerID	customer_last_na...	customer_first_na...	customer_email	customer_pho...
1	saxena	kriti	kesaxena@gmail.com	805-805-1357
2	lee	joseph	jlee932cp@gmail.com	213-276-2724
3	pelham	kyle	kyp123@gmail.com	699-123-4567
4	chiv	andy	andy204@gmail.com	805-750-3461
5	zhan	yong	zyong@hotmail.com	805-124-5478
NULL	NULL	NULL	NULL	NULL

```
162 •  select*from Payments;
163
```

100% 5:162

Result Grid Filter Rows: Search Edit: Export/Import:

paymentID	payment_meth...	customerID
101	apple_pay	1
102	google_pay	2
103	card	3
104	card	4
105	paypal	5
106	paypal	4
107	google_pay	4
108	apple_pay	2
NULL	NULL	NULL

```
168 •  select * from googlepay;
```

100% 1:170

Result Grid Filter Rows: Search Edit: Export/Import:

googlepayID	google_pay_email
101	jlee932cp@gmail.com
107	andy204@gmail.com
NULL	NULL

```
174 •  select * from applepay;
175    # card
176
```

100% 24:174

Result Grid Filter Rows: Search Edit: Export/Import:

apple_payID	apple_pay_email
101	kesaxena@gmail.com
108	jlee932cp@gmail.com
NULL	NULL

```
180 •  select * from card;
```

100% 20:180

Result Grid Filter Rows: Search Edit: Export/Import:

cardID	card_number
103	4426888877776666
104	4111222233334444
NULL	NULL

```
185 •  select * from paypal;
```

100% 1:187

Result Grid Filter Rows: Search Edit: Export/Import:

paypalID	paypal_numb...
105	805-124-5478
106	805-750-3461
NULL	NULL

```
198 •  select*from Deliveries;
```

100% 6:198

Result Grid Filter Rows: Search Edit: Export/Import:

deliveryID	delivery_meth...
1001	selfpickup
1002	uber
1003	doordash
1004	selfpickup
1005	doordash
1006	doordash
1007	uber
1008	doordash
NULL	NULL

```
208 •  select*from Doordash;
```

100% 4:208

Result Grid Filter Rows: Search Edit: Export/Import:

doordashID	driver_name	driver_phone	pick_up_time	delivery_time
1003	Daniel Ric	3057041345	2022-04-24 11:00:00	2022-04-24 11:20:00
1005	Karl Max	6993051257	2020-03-10 18:00:00	2020-03-10 18:25:00
1006	Vance Miller	8468885555	2020-03-13 15:00:00	2020-03-13 15:20:00
1008	Lewis Hamilton	8468345555	2020-10-10 19:00:00	2020-10-10 20:00:00
NULL	NULL	NULL	NULL	NULL

```
214 •  select*from Uber;
```

100% 7:214

Result Grid Filter Rows: Search Edit: Export/Import:

uberID	driver_name	driver_phone	pick_up_time	delivery_time
1002	John Smith	8052224444	2021-11-20 16:15:00	2021-11-20 16:30:00
1007	Larry Elder	8054567890	2021-04-24 12:30:00	2021-04-24 12:45:00
NULL	NULL	NULL	NULL	NULL

```
221 •  select * from selfpickup;
```

100% 26:221

Result Grid Filter Rows: Search Edit: Export/Import:

pickupID	pick_up_time
1001	2022-01-02 12:15:00
1004	2022-03-13 11:00:00
NULL	NULL

```
232 •  select*from Orders;
233
```

100% 11:232

Result Grid Filter Rows: Search Edit: Export/Import:

orderID	order_date	paymentID	customerID	deliveryID
101	2021-11-20	101	1	1001
102	2022-01-02	102	2	1002
103	2022-04-02	103	3	1003
104	2021-04-24	104	4	1004
105	2020-03-10	105	5	1005
106	2020-03-13	106	4	1006
107	2022-03-13	107	4	1007
108	2020-10-10	108	2	1008
NULL	NULL	NULL	NULL	NULL

```
244 •  select*from Popup;
245
```

100% 6:244

Result Grid Filter Rows: Search Edit: Export/Import:

popupID	popup_name	rating	popup_email
11	Floating bar	4.8	customercare@fbar.com
22	Frank's Frankfurters	5	customercare@ff.com
33	Thai Palace	4.5	customercare@tp.com
44	Spaghetti Factory	5	customercare@sf.com
55	Mel's Diner	3.5	customercare@md.com
66	Fredy's BBQ	3	customercarefbbq@gmail.com
77	Papa's kitchen	4.2	customercare@pk.com
88	Dhaba 65	5	customercareD65@gmail.com
NULL	NULL	NULL	NULL

```
257 •  select*from Food;
258
```

100% 4:257

Result Grid Filter Rows: Search Edit: Export/Import:

foodID	food_name	food_description	country
111	tacos	with beef and extra onions	Mexico
222	hotdog	hot dog without mustard	US
333	pad thai	chicken pad thai with green onions	Thailand
444	spaghetti	spaghetti and meatballs	Italy
555	hamburger	double burger with cheese	US
666	southern style bbq ribs	extra sauce	US
777	Korean fried chicken	Extra spicy	Korean
888	Chicken tikka masala	with garlic butter naan	India
NULL	NULL	NULL	NULL

```
274 •  select*from Popup_food;
275
```

100% 13:274

Result Grid Filter Rows: Search Edit: Export/Import:

popup_foodID	priceEach	popupID	foodID
1	10	11	111
2	15	11	222
3	12	22	444
4	11	33	555
5	16	33	666
6	10	33	777
7	9	44	888
8	15	55	111
9	13	66	111
10	13	77	444
11	10	88	333
NULL	NULL	NULL	NULL

```
290 •  select*from Popup_food_orders;
291
```

100% 8:290

Result Grid Filter Rows: Search Edit: Export/Import:

popup_food_orde...	quantity	popup_foodID	orderID
1	3	1	101
2	2	2	101
3	2	4	102
4	1	5	102
5	5	6	102
6	5	3	103
7	4	7	104
8	2	8	105
9	8	9	106
10	9	10	107
11	5	11	108
NULL	NULL	NULL	NULL

8. Examples of SQL Queries for Business Insight

Query 1: Across all popups, which food was the most ordered? Display food name, food ID and number of time the food is ordered

```
SELECT pf.foodID, f.food_name, count(*) AS num_ordered
FROM popup_food_orders pfo JOIN popup_food pf JOIN food f
ON pfo.popup_foodID = pf.popup_foodID AND pf.foodID = f.foodID
GROUP BY pf.foodID
ORDER BY num_ordered DESC LIMIT 1;
```

foodID	food_name	num_ordered
111	tacos	3

Query 2: Find all orders delivered by Doordash and order by order date.

```
SELECT *
FROM orders o JOIN deliveries d
USING(deliveryID)
WHERE delivery_method = "doordash"
ORDER BY order_date ASC;
```

deliveryID	orderID	order_date	paymentID	customerID	delivery_meth...
1005	105	2020-03-10	105	5	doordash
1006	106	2020-03-13	106	4	doordash
1008	108	2020-10-10	108	2	doordash
1003	103	2022-04-02	103	3	doordash

Query 3: Calculate all revenues for each popup

```
SELECT p.popupID, p.popup_name, SUM(pf.priceEach*pfo.quantity) AS total_revenue
FROM popup_food_orders pfo JOIN popup_food pf JOIN popup p
ON pfo.popup_foodID = pf.popup_foodID AND pf.popupID = p.popupID
GROUP BY p.popupID, p.popup_name
ORDER BY total_revenue DESC;
```

popupID	popup_name	total_revenue
77	Papa's kitchen	117
66	Fredy's BBQ	104
33	Thai Palace	88
11	Floating bar	60
22	Frank's Frankfurters	60
88	Dhaba 65	50
44	Spaghetti Factory	36
55	Mel's Diner	30

Query 4: Find the average price for every food

```
SELECT f.foodID, f.food_name, FORMAT(AVG(pf.priceEach),2) AS avg_price
FROM popup_food pf JOIN food f
ON pf.foodID = f.foodID
GROUP BY f.foodID
ORDER BY avg_price DESC;
```

foodID	food_name	avg_price
888	Chicken tikka masala	9.00
666	southern style bbq ribs	16.00
222	hotdog	15.00
111	tacos	12.67
444	spaghetti	12.50
555	hamburger	11.00
333	pad thai	10.00
777	Korean fried chicken	10.00

Query 5: Find the highest cost of food order for every payment method

```
SELECT t.payment_method, max(t.total_cost) AS highest_cost
FROM
(SELECT pfo.orderID, p.payment_method, SUM(pf.priceEach*pfo.quantity) AS total_cost
FROM orders o JOIN deliveries d JOIN popup_food_orders pfo JOIN popup_food pf JOIN
payments p
ON o.deliveryID = d.deliveryID AND o.orderID = pfo.orderID AND pfo.popup_foodID =
pf.popup_foodID AND o.paymentID = p.paymentID
GROUP BY pfo.orderID) T
GROUP BY t.payment_method
ORDER BY highest_cost DESC;
```

payment_method	highest_co...
google_pay	117
paypal	104
apple_pay	60
card	60