

Perfect Pizza Operation System Design

Table of Contents

[Executive Summary 2](#_Toc132547269)

[System Initiation 3](#_Toc132547270)

[Data Flow Diagram (DFD) 3](#_Toc132547271)

[List of entities 3](#_Toc132547272)

[List of Primary and Foreign keys 5](#_Toc132547273)

[SQL tables 5](#_Toc132547274)

[Database SQL Tables 5](#_Toc132547275)

[Trigger 7](#_Toc132547276)

[Trigger test Demo 8](#_Toc132547277)

[Populating table 8](#_Toc132547278)

[Customer table 8](#_Toc132547279)

[Orders table 9](#_Toc132547280)

[cook table 9](#_Toc132547281)

[Food table 10](#_Toc132547282)

[Delivery table 10](#_Toc132547283)

[Management table 10](#_Toc132547284)

[System design 11](#_Toc132547285)

[Entity Relational Diagram (ERD) 11](#_Toc132547286)

[Relational Diagram 12](#_Toc132547287)

[Retrieved tables 12](#_Toc132547288)

[a) Retrieve the profile of a customer 12](#_Toc132547289)

[b) Retrieve the Cus\_Name, Cus\_Address, Product, Total Amount, Tax, Delivery charge, and Driver id. 13](#_Toc132547290)

[c) Retrieve all the drivers that made deliveries this week from the Delivery table. 13](#_Toc132547291)

[d) Retrieve Management profile and Weekly total. Etc…. 14](#_Toc132547292)

# **Executive Summary**

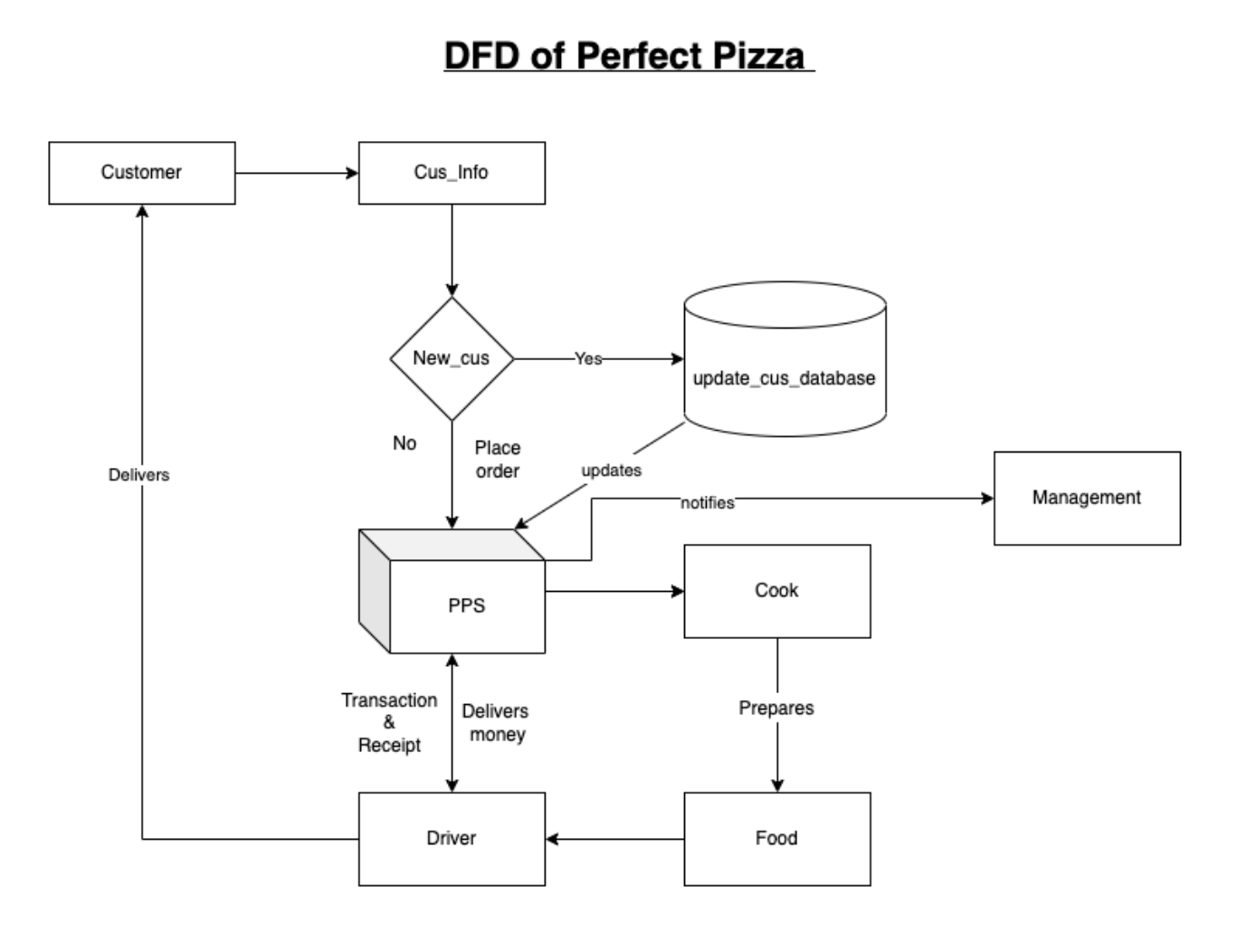
The pizza chain called Perfect Pizza wants to set up a system to track pizza and Buffalo wing orders. After entering a regular customer's phone number into the computer, the system should be able to retrieve the customer’s name, address, and the date of their last purchase. Additionally, the system should be able to compute the final cost including tax and delivery cost and send the order to the cook. A receipt and, and sometimes unique offers for customers to receive discounts should be printed. Customers receive a duplicate of the receipt and coupon from delivery drivers (if any is applicable to them). The Management keeps weekly totals for data comparison with the results from the previous year.

To do this, a data flow diagram (DFD) that shows the movement of processes and the activities within Perfect Pizza will be developed. To create the data structures in the design, all potential entities and their corresponding attributes will be enumerated. Each entity will show primary keys and foreign keys when appropriate. The required database tables will be built, designed, and filled with data.

An entity-relationship diagram (ERD) and a relational diagram will be created from the database design to show the connections between the design's entities. When a customer calls to place an order, SQL queries will be written to obtain the customer's profile, the customer's name, address, product, total amount, tax, delivery charge, and driver ID, as well as the drivers who made deliveries this week, the Management profile, and the weekly total.

# **System Initiation**

# Data Flow Diagram (DFD)



## List of entities

The entities and their respective attributes that make up the data structures in the design are:

1. **Customer entity:**

* cus\_ID (primary key)
* cus\_name
* cus\_phone\_N
* cus\_address
* cus\_Pizza\_name
* cus\_lastOrder\_date

1. **Orders entity:**

* order\_ID (primary key)
* cus\_ID (foreign key referencing Customer entity)
* order\_address
* order\_phone\_N
* order\_Date
* Total
* Tax
* delivery\_Cost
* driver\_ID (foreign key referencing Delivery entity)

1. **Cook entity:**

* cook\_ID (primary key)
* order\_ID (foreign key referencing Orders entity)
* cook\_name
* cook\_specialty

1. **Food entity:**

* food\_ID (primary key)
* cook\_ID (foreign key referencing Cook entity)
* food\_name
* food\_ingr
* food\_price

1. **Delivery entity:**

* driver\_ID (primary key)
* driver\_name
* delivery\_date
* receipt\_ID
* coupon\_ID
* order\_id (foreign key referencing Orders entity)
* food\_ID (foreign key referencing Food entity)
* payment

1. **Management entity:**

* manager\_ID (primary key)
* manager\_name
* weekly\_total
* driver\_ID (foreign key referencing Delivery entity)

1. **PPS entity:**

* PPS\_ID (primary key)
* cus\_ID (foreign key referencing Customer entity)
* order\_ID (foreign key referencing Orders entity)
* cook\_ID (foreign key referencing Cook entity)
* food\_ID (foreign key referencing Food entity)
* driver\_ID (foreign key referencing Delivery entity)
* manager\_ID (foreign key referencing Management entity)

## List of Primary and Foreign keysTop of Form

Bottom of Form

* **Customer:**
  + Primary Key: cus\_ID
* **Orders:**
  + Primary Key: order\_ID
  + Foreign Key: cus\_ID (references Customer(cus\_ID))
  + Foreign Key: driver\_ID ( referencing Delivery (driver\_ID))
* **Cook:**
  + Primary Key: cook\_ID
  + Foreign Key: order\_ID (references Orders(order\_ID))
* **Food:**
  + Primary Key: food\_ID
  + Foreign Key: cook\_ID (references Cook(cook\_ID))
* **Delivery:**
  + Primary Key: driver\_ID
  + Foreign Key: order\_ID (references Orders(order\_ID))
  + Foreign Key: food\_ID (references Food(food\_ID))
* **Management:**
  + Primary Key: manager\_ID
  + Foreign Key: driver\_ID (references Delivery(driver\_ID))
* **PPS:**
  + Primary Key: PPS\_ID
  + Foreign Key: cus\_ID (references Customer(cus\_ID))
  + Foreign Key: order\_ID (references Orders(order\_ID))
  + Foreign Key: cook\_ID (references Cook(cook\_ID))
  + Foreign Key: food\_ID (references Food(food\_ID))
  + Foreign Key: driver\_ID (references Delivery(driver\_ID))
  + Foreign Key: manager\_ID (references Management(manager\_ID))

Top of Form

Bottom of Form

# **SQL tables**

## Database SQL Tables

CREATE TABLE Customer (

cus\_ID INT AUTO\_INCREMENT NOT NULL,

cus\_name VARCHAR(50) NOT NULL,

cus\_phone\_N VARCHAR(20) NOT NULL,

cus\_address VARCHAR(50) NOT NULL,

cus\_Pizza\_name VARCHAR(50) NOT NULL,

cus\_lastOrder\_date DATE NOT NULL,

PRIMARY KEY (cus\_ID)

);

CREATE TABLE Orders (

order\_ID INT NOT NULL,

cus\_ID INT NOT NULL,

order\_address VARCHAR(50) NOT NULL,

order\_phone\_N VARCHAR(20) NOT NULL,

order\_Date DATE NOT NULL,

Total DECIMAL(15, 2) NOT NULL,

Tax DECIMAL(10, 2) NOT NULL,

delivery\_Cost DECIMAL(20, 2) NOT NULL,

driver\_ID INT NOT NULL,

PRIMARY KEY (order\_ID),

FOREIGN KEY (cus\_ID) REFERENCES Customer(cus\_ID)

);

CREATE TABLE Cook (

cook\_ID INT NOT NULL,

order\_ID INT NOT NULL,

cook\_name VARCHAR(50) NOT NULL,

cook\_specialty VARCHAR(50) NOT NULL,

PRIMARY KEY (cook\_ID),

FOREIGN KEY (order\_ID) REFERENCES Orders(order\_ID)

);

CREATE TABLE Food (

food\_ID INT NOT NULL,

cook\_ID INT NOT NULL,

food\_name VARCHAR (50) NOT NULL,

food\_ingr VARCHAR (100) NOT NULL,

food\_price DECIMAL(15,2) NOT NULL,

PRIMARY KEY (food\_ID),

FOREIGN KEY (cook\_ID) REFERENCES Cook(cook\_ID)

);

CREATE TABLE Delivery (

driver\_ID INT NOT NULL,

driver\_name VARCHAR(50) NOT NULL,

delivery\_date DATE NOT NULL,

receipt\_ID INT NOT NULL,

coupon\_ID INT,

order\_id INT NOT NULL,

food\_ID INT NOT NULL,

payment DECIMAL(10, 2) NOT NULL,

PRIMARY KEY (driver\_ID),

FOREIGN KEY (order\_ID) REFERENCES orders (order\_ID),

FOREIGN KEY (food\_ID) REFERENCES Food(food\_ID)

);

CREATE TABLE Management (

manager\_ID INT,

manager\_name VARCHAR(50),

weekly\_total DECIMAL(10,2),

driver\_ID INT NOT NULL,

PRIMARY KEY (manager\_ID),

FOREIGN KEY (driver\_ID) REFERENCES Delivery(driver\_ID)

);

CREATE TABLE PPS(

PPS\_ID INT NOT NULL,

cus\_ID INT NOT NULL,

order\_ID INT NOT NULL,

cook\_ID INT NOT NULL,

food\_ID INT NOT NULL,

driver\_ID INT NOT NULL,

manager\_ID INT NOT NULL,

PRIMARY KEY (PPS\_ID),

FOREIGN KEY (cus\_ID) REFERENCES Customer(cus\_ID),

FOREIGN KEY (order\_ID) REFERENCES Orders(order\_ID),

FOREIGN KEY (cook\_ID) REFERENCES Cook(cook\_ID),

FOREIGN KEY (food\_ID) REFERENCES Food(food\_ID),

FOREIGN KEY (driver\_ID) REFERENCES Delivery (driver\_ID),

FOREIGN KEY (manager\_ID) REFERENCES Management (manager\_ID)

);

### **Trigger**

DELIMITER $$

CREATE TRIGGER Customer\_Order\_Trigger

BEFORE INSERT ON Orders

FOR EACH ROW

BEGIN

IF NEW.cus\_ID IS NULL THEN

INSERT INTO Customer (cus\_address, cus\_phone\_N, cus\_lastOrder\_date)

VALUES (NEW.order\_address, NEW.order\_phone\_N, NEW.order\_Date);

SET NEW.cus\_ID = LAST\_INSERT\_ID();

END IF;

END$$

DELIMITER ;

### **Trigger test Demo**

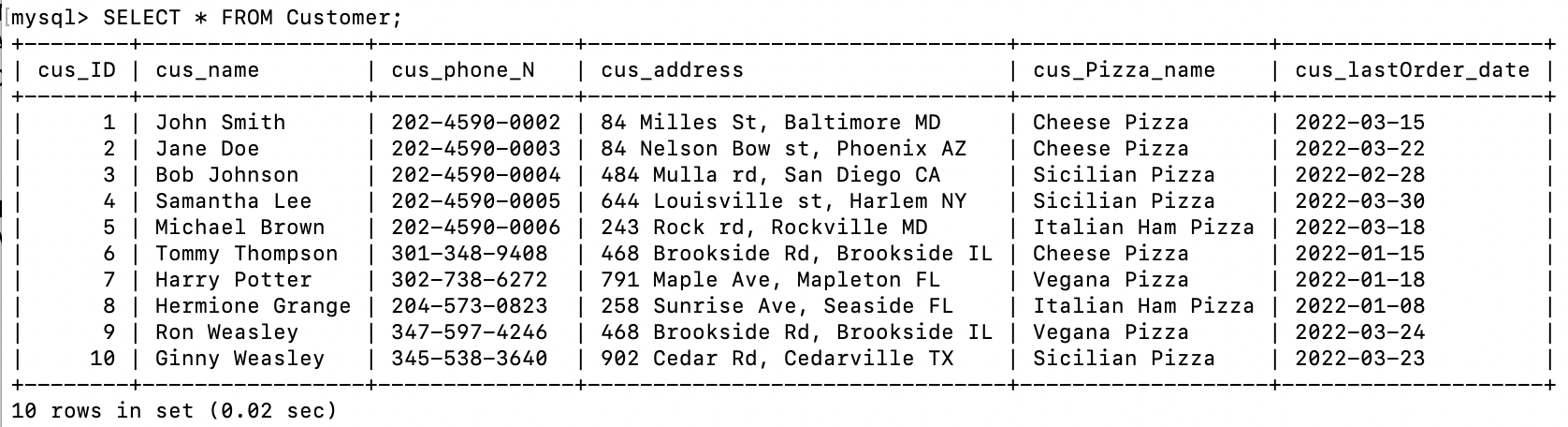
**Insert Code**

INSERT INTO Customer (cus\_ID, cus\_name, cus\_phone\_N, cus\_address, cus\_Pizza\_name, cus\_lastOrder\_date)

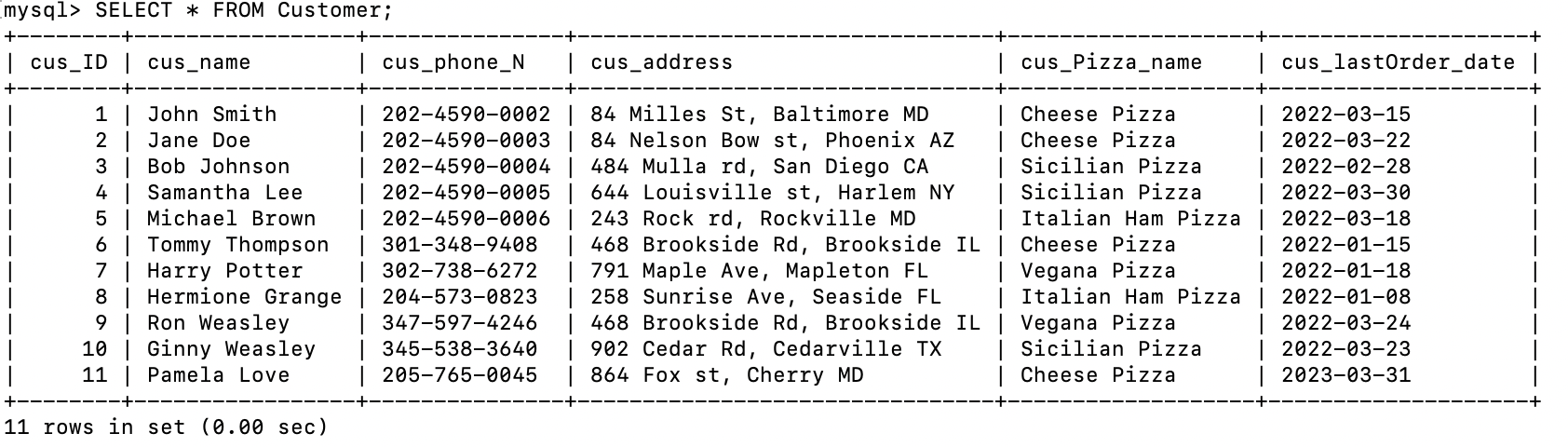
VALUES

(NULL, "Pamela Love ", "205-765-0045", "864 Fox st, Cherry MD","Cheese Pizza", "2023-03-31");

**Table Before**

****

**Table After Trigger**

****

## Populating table

### Customer table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **cus\_ID** | **cus\_name** | **cus\_phone\_N** | **cus\_address** | **cus\_Pizza\_name** | **cus\_lastOrder\_date** |
| **1** | John Smith | 202-4590-0002 | 84 Milles St, Baltimore MD | Cheese Pizza | 2022-03-15 |
| **2** | Jane Doe | 202-4590-0003 | 84 Nelson Bow st, Phoenix AZ | Cheese Pizza | 2022-03-22 |
| **3** | Bob Johnson | 202-4590-0004 | 484 Mulla rd, San Diego CA | Sicilian Pizza | 2022-02-28 |
| **4** | Samantha Lee | 202-4590-0005 | 644 Louisville st, Harlem NY | Sicilian Pizza | 2022-03-30 |
| **5** | Michael Brown | 202-4590-0006 | 243 Rock rd, Rockville MD | Italian Ham Pizza | 2022-03-18 |
| **6** | Tommy Thompson | 301-348-9408 | 468 Brookside Rd, Brookside IL | Cheese Pizza | 2022-01-15 |
| **7** | Harry Potter | 302-738-6272 | 791 Maple Ave, Mapleton FL | Vegana Pizza | 2022-01-18 |
| **8** | Hermione Grange | 204-573-0823 | 258 Sunrise Ave, Seaside FL | Italian Ham Pizza | 2022-01-08 |
| **9** | Ron Weasley | 347-597-4246 | 468 Brookside Rd, Brookside IL | Vegana Pizza | 2022-03-24 |
| **10** | Ginny Weasley | 345-538-3640 | 902 Cedar Rd, Cedarville TX | Sicilian Pizza | 2022-03-23 |

### Orders table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **order\_ID** | **cus\_ID** | **order\_address** | **order\_phone\_N** | **order\_Date** | **Total** | **Tax** | **delivery\_Cost** | **driver\_ID** |
| **1** | 3 | 484 Mulla rd, San Diego CA | 202-4590-0004 | 2022-02-28 | 50.00 | 3.50 | 5.00 | 2 |
| **2** | 6 | 468 Brookside Rd, Brookside IL | 301-348-9408 | 2022-01-15 | 200.00 | 14.00 | 12.00 | 6 |
| **3** | 1 | 84 Milles St, Baltimore MD | 202-4590-0002 | 2022-03-15 | 30.00 | 7.00 | 8.00 | 1 |
| **4** | 2 | 84 Nelson Bow st, Phoenix AZ | 202-4590-0003 | 2022-03-22 | 25.00 | 3.75 | 3.00 | 3 |
| **5** | 9 | 468 Brookside Rd, Brookside IL | 347-597-4246 | 2022-03-24 | 15.00 | 5.50 | 9.00 | 5 |
| **6** | 8 | 258 Sunrise Ave, Seaside FL | 204-573-0823 | 2022-01-15 | 20.00 | 4.00 | 12.00 | 6 |
| **7** | 5 | 243 Rock rd, Rockville MD | 202-4590-0006 | 2022-03-18 | 25.00 | 8.75 | 10.00 | 2 |
| **8** | 4 | 644 Louisville st, Harlem NY | 202-4590-0005 | 2022-03-30 | 30.00 | 3.50 | 5.00 | 1 |
| **9** | 7 | 791 Maple Ave, Mapleton FL | 302-738-6272 | 2022-01-18 | 35.00 | 5.25 | 6.00 | 4 |
| **10** | 2 | 902 Cedar Rd, Cedarville TX | 345-538-3640 | 2022-03-23 | 15.00 | 7.00 | 8.00 | 3 |

### cook table

|  |  |  |  |
| --- | --- | --- | --- |
| **cook\_ID** | **order\_ID** | **cook\_name** | **cook\_specialty** |
| **1** | 1 | John Smith | Italian Pizza |
| **2** | 2 | Maria Rodriguez | Mexican Pizza Style |
| **3** | 3 | David Lee | Italian Pizza |
| **4** | 4 | Fatima Khan | Mexican Pizza Style |
| **5** | 5 | Alexandre Dupont | French Pizza |
| **6** | 6 | Marta Silva | Italian Pizza |
| **7** | 7 | Andrea Rossi | New York Pizza Style |
| **8** | 8 | Oliver Schmidt | Spanish Pizza Style |
| **9** | 9 | Aisha Ahmed | Mediterian Pizza Style |
| **10** | 10 | Javier Hernandez | Spanish Pizza Style |

### 

### Food table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **food\_ID** | **cook\_ID** | **food\_name** | **food\_ingr** | **food\_price** |
| **1** | 1 | Cheese Pizza | Dough, tomato sauce, mozzarella cheese | 10.99 |
| **2** | 2 | Sicilian Pizza | Dough, tomato sauce, parmesan cheese, oregano | 12.99 |
| **3** | 3 | Italian Ham Pizza | Dough, tomato sauce, ham, mozzarella cheese, parmesan cheese | 14.99 |
| **4** | 4 | Vegana Pizza | Dough, tomato sauce, mushrooms, green peppers, onions, olives, artichokes | 15.99 |

### Delivery table

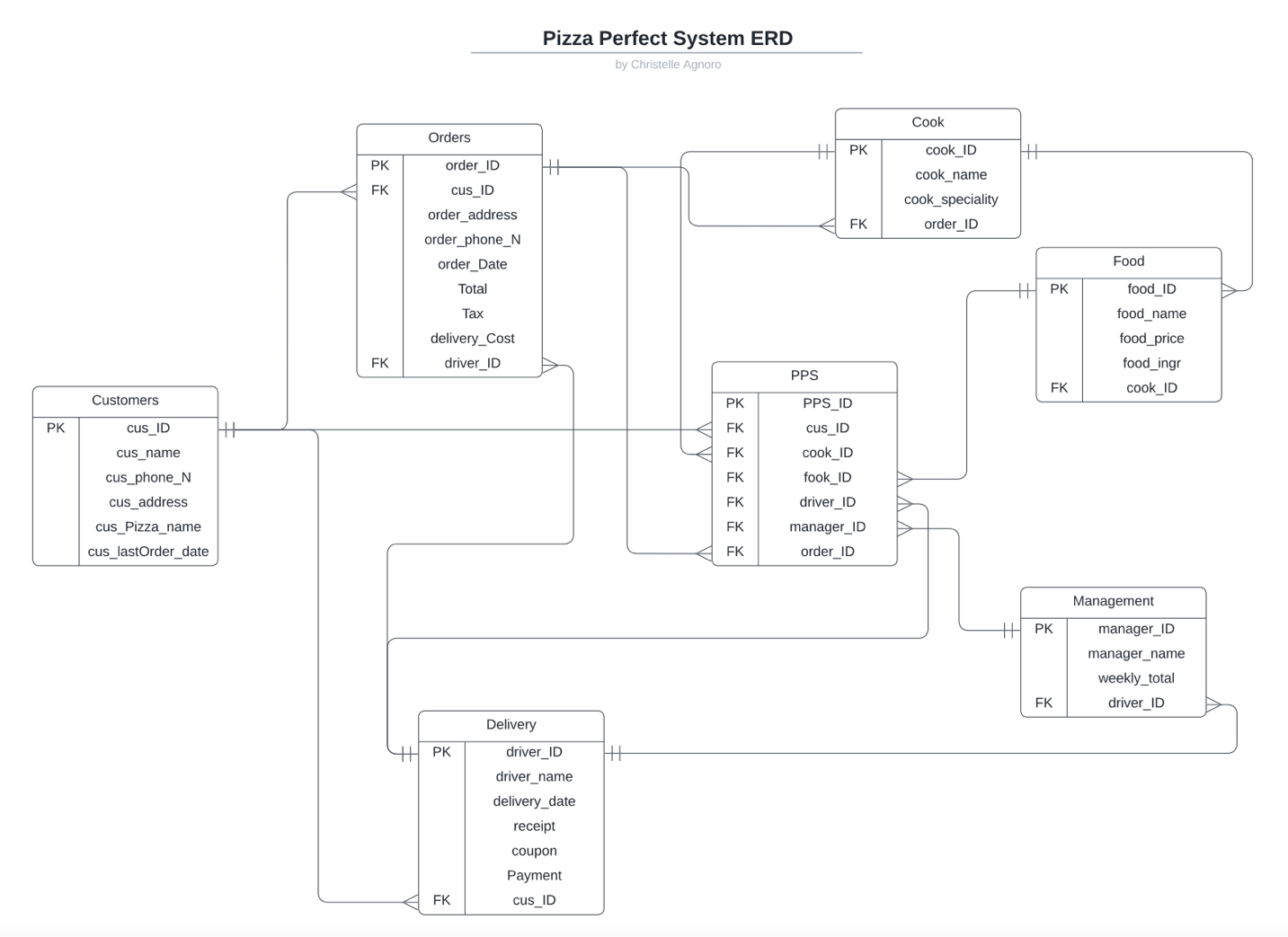
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **driver\_ID** | **driver\_name** | **delivery\_date** | **receipt\_ID** | **coupon\_ID** | **order\_id** | **food\_ID** | **payment** |
| **1** | John Smith | 2023-03-28 | 12345 | NULL | 1 | 1 | 20.99 |
| **2** | Emily Jones | 2023-03-29 | 12346 | NULL | 2 | 1 | 15.99 |
| **3** | Mark Thompson | 2023-03-30 | 12347 | 123 | 3 | 2 | 30.25 |
| **4** | Samantha Davis | 2023-03-31 | 12348 | 124 | 4 | 2 | 18.00 |
| **5** | David Rodriguez | 2023-04-01 | 12349 | 125 | 5 | 3 | 20.00 |
| **6** | Karen Lee | 2023-04-02 | 12350 | NULL | 6 | 3 | 14.30 |
| **7** | Juan Hernandez | 2023-04-03 | 12351 | 127 | 7 | 4 | 23.94 |
| **8** | Jessica Nguyen | 2023-04-04 | 12352 | NULL | 8 | 4 | 17.99 |
| **9** | William Kim | 2023-04-05 | 12353 | 129 | 9 | 1 | 38.74 |
| **10** | Linda Wilson | 2023-04-06 | 12354 | 789 | 10 | 2 | 14.68 |

### Management table

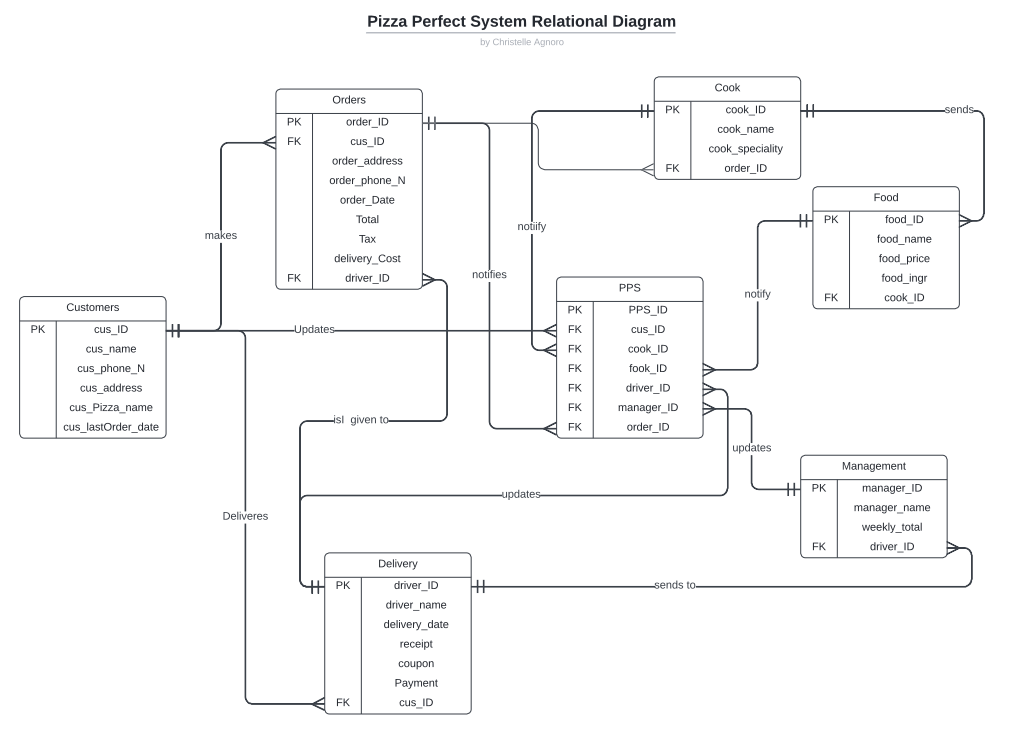
|  |  |  |  |
| --- | --- | --- | --- |
| **manager\_ID** | **manager\_name** | **weekly\_total** | **driver\_ID** |
| **1** | John Smith | 1250.50 | 1 |
| **2** | Jane Doe | 1100.75 | 2 |
| **3** | Mike Johnson | 975.25 | 3 |
| **4** | Sarah Lee | 1850.00 | 4 |
| **5** | David Kim | 1200.00 | 5 |

# **System design**

## Entity Relational Diagram (ERD)



## Relational Diagram



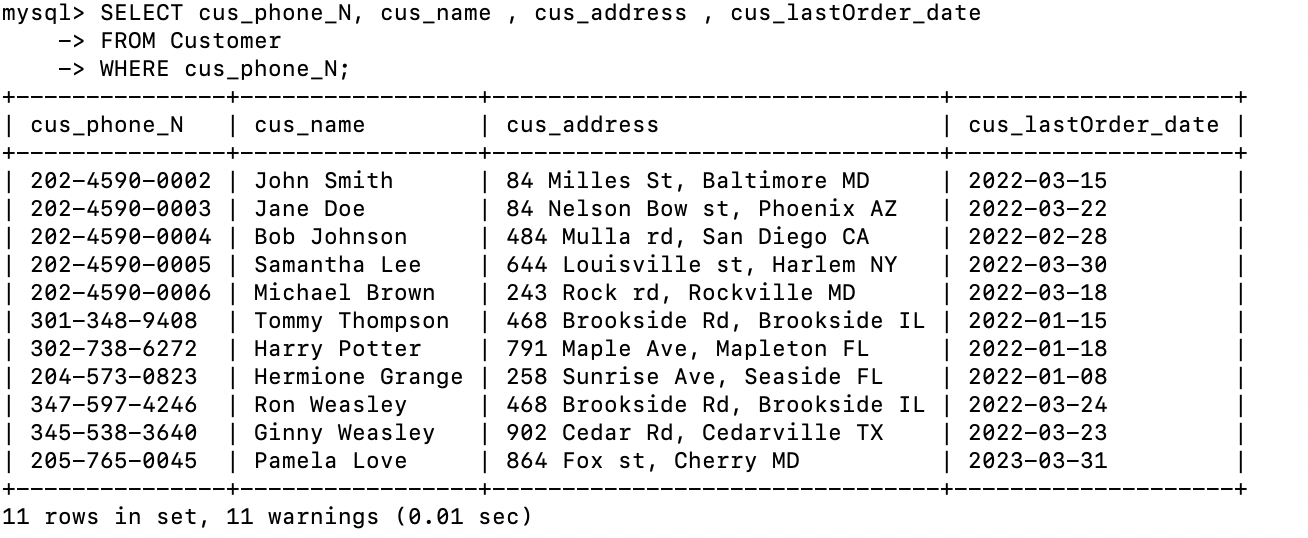
## Retrieved tables

### Retrieve the profile of a customer

SELECT cus\_phone\_N, cus\_name , cus\_address , cus\_lastOrder\_date

FROM Customer

WHERE cus\_phone\_N;



### Retrieve the Cus\_Name, Cus\_Address, Product, Total Amount, Tax, Delivery charge, and Driver id.

SELECT cus\_name ,o.order\_address , f.food\_name , o.Total, o.Tax, o.delivery\_Cost , d.driver\_ID

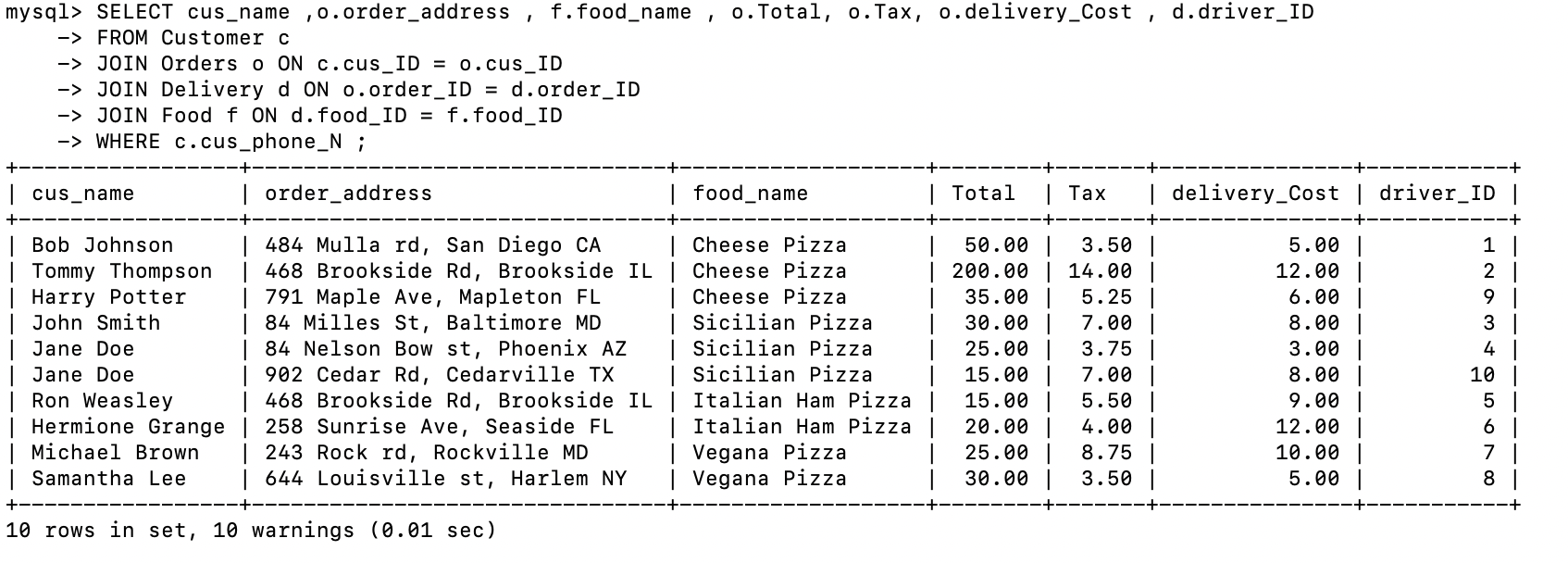
FROM Customer c

JOIN Orders o ON c.cus\_ID = o.cus\_ID

JOIN Delivery d ON o.order\_ID = d.order\_ID

JOIN Food f ON d.food\_ID = f.food\_ID

WHERE c.cus\_phone\_N ;

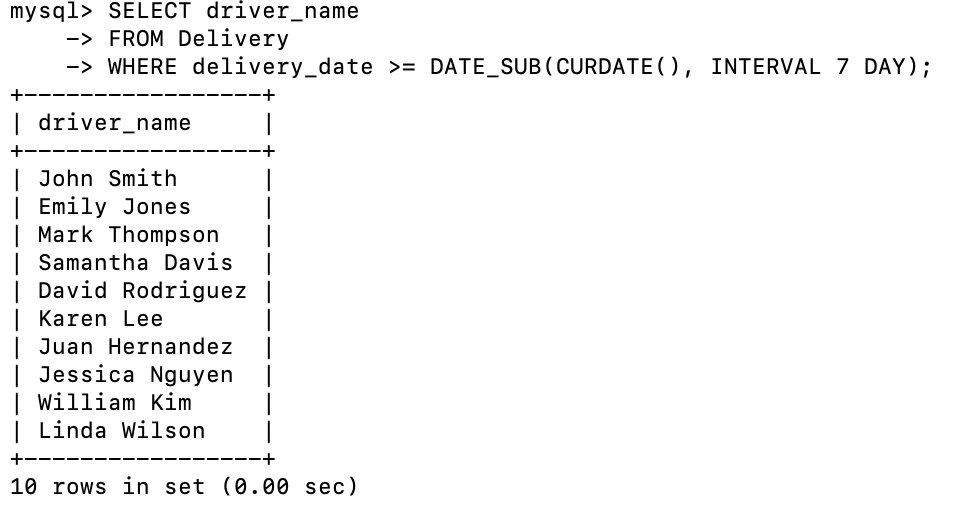


### Retrieve all the drivers that made deliveries this week from the Delivery table.

SELECT driver\_name

FROM Delivery

WHERE delivery\_date >= DATE\_SUB(CURDATE(), INTERVAL 7 DAY);



### Retrieve Management profile and Weekly total. Etc….

SELECT manager\_ID, manager\_name, weekly\_total

FROM Management;

