

Restaurant Transactional Database

Mikaël Guillin
TY Kang
Jeonghoon Lee
Quasim Simmons
Mohammad Alam

I. Business Scenario

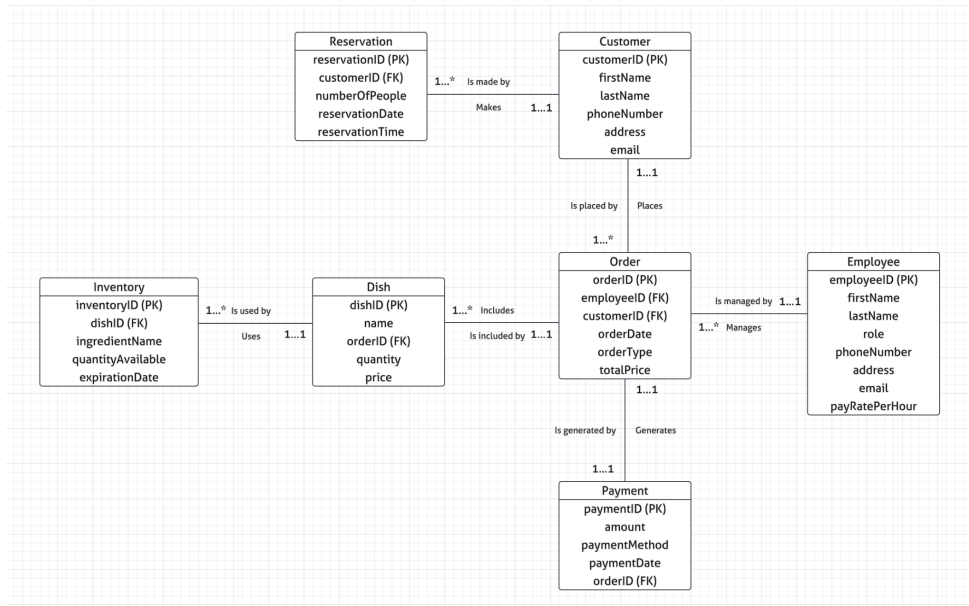
The Restaurant operates on a membership basis, requiring customers to register before making walk-in visits or reservations. Customers provide their information, including Name, Address, Phone number, and E-mail. Reservations are managed using customer details, schedules, and the number of guests.

When visiting, customers place orders as either takeaway or dine-in. Each order comprises one or more dishes selected from the restaurant's menu. Payments are processed based on the ordered dishes, considering their price and quantity. The order, dish, and payment information are linked to ensure accurate billing and record-keeping. Additionally, there is also an inventory management system to monitor ingredient stock levels and expiration dates, ensuring essential items are adequately supplied.

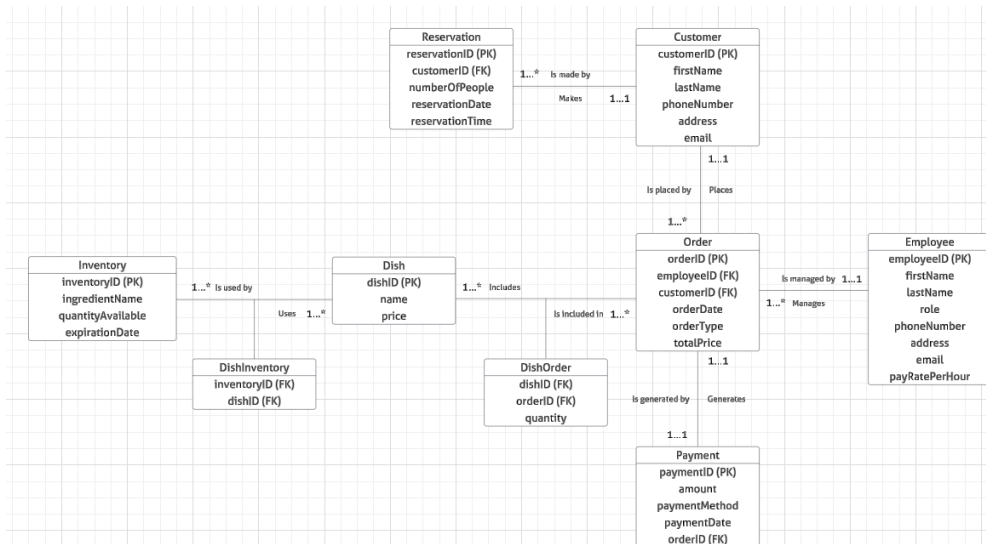
Employee information is managed using basic details like personal info, job roles, and pay rates. For operational efficiency, every order or reservation is assigned to a specific employee who handles it from start to finish. This includes confirming reservations, ensuring the order is prepared accurately, delivering the order to the customer, and processing the payment.

The restaurant's database integrates customer management, order and payment tracking, inventory control, and employee task assignments. This system enhances overall efficiency and provides clear visibility into the business's operations.

II. ER Model



[version 1]



[version 2]

Relationship Sentences

- One Reservation may be made by one to many Customers.
- One Customer makes one and only one Reservation.
- One Customer may place one to many Orders.
- One Order is placed by one and only one Customer.
- One Order is managed by one and only one Employee.
- One Employee may manage one to many Orders.
- One Order may include one to many Dishes.
- One Dish may be included in one to many Orders.
- One Order generates one and only one Payment.
- One Payment is generated by one and only one Order.
- One Dish may use one to many Inventory.
- One Inventory may be used in one to many Dishes.

III. Conversion to Relational Model

Reservation

(reservationID(pk), customerID(fk), numberOfPeople, reservationDate, reservationTime)

Customer

(customerID(pk), firstName, lastName, phoneNumber, address, email)

Order

(orderID(pk), employeeID(fk), customer(fk), orderDate, orderType, totalPrice)

Employee

(employeeID(pk), firstName, lastName, role, phoneNumber, address, email, payRatePerHour)

Payment

(paymentID(pk), amount, paymentMethod, paymentDate, orderID(fk))

Dish

(dishID(pk), name, price)

DishOrder

(dishID(fk), orderID(fk), quantity)

DishInventory

(dishID(fk), inventoryID(fk))

Inventory

(inventoryID(pk), ingredientName, quantityAvailable, expirationDate)

IV. Normalization

Customer table

customerID	firstName	lastName	phoneNumber	address	email
1	Kathryn	Jacobson	(744) 577-6982	47652 Chesley Knolls Jersey City New Jersey 07302	Kathryn.Jacob son@hotmail.c om
2	Garnet	Yost	(235) 906-3425	3626 Brenda Circle Brooklyn New York	Garnet.Yost@ hotmail.com

1NF: Does not satisfy 1NF

Solution: split address attribute into street, city, state, zipCode atomic attributes

2NF: Satisfies 2NF, no partial key dependencies

3NF: Transitive dependency exists, customerID -> zipCode, zipCode -> city, state.

Solution: Make a new table named "Location" and replace "address" by a zipCode attribute used as a foreign key in Customer table.

Location table

zipCode	city	state
07302	Jersey City	New Jersey
11201	Brooklyn	New York

Employee table

employeeID	firstName	lastName	phoneNumber	address	role	payRate	email
1	Triston	Boyer	(387) 304-7358	3167 Castle Road Jersey City New Jersey 07302	Manager	28	Triston_Boyer@yahoo.com
2	Maximillian	Parisian	(850) 881-0753	51947 Kozey Skyway Brooklyn New York	Chef	20	Maximillian.Parisian@yahoo.com

1NF: Does not satisfy 1NF

Solution: split address into street, city, state, zipCode atomic attributes

2NF: No partial key dependencies

3NF: Transitive dependency exists, employeeID → zipCode, zipCode → city, state.

Solution: Make a new table named "Location" (See example above)

Order table

orderID	employeeID	customerID	orderDate	orderType	totalPrice
1	1	1	2024-02-18 00:00:00	Dine-In	112
2	1	2	2023-12-23 00:00:00	Takeout	32

1NF: Satisfies 1NF

2NF: Satisfies 2NF, no partial key dependencies

3NF: Satisfies 3NF, no transitive keys

Dish table

dishID	orderID	name	quantity	price
1	1	Pork Belly Buns	2	5
2	2	Nashi Pear And Pear Tart	1	13

1NF: Satisfies 1NF

2NF: Does not satisfy 2NF since name and price attributes depend only on dishID and not on the full composite key dishID + orderID.

Solution: Remove orderID attribute and make a new table named OrderDish that includes dishID and orderID as foreign keys and the quantity attribute.

OrderDish table

dishID	orderID	quantity
1	1	2
2	2	1

3NF: Satisfies 3NF, no transitive dependencies

Inventory table

inventoryID	dishID	ingredientName	quantityAvailable	expirationDate
1	1	jicama	32	2025-11-12 00:00:00
2	2	passionfruit	38	2025-05-22 00:00:00

1NF: Satisfies 1NF

2NF: Does not satisfy 2NF since ingredientName, quantityAvailable, expirationDate depend only on inventoryID and not on the full composite key inventoryID + dishID.

Solution: Remove dishID attribute and make a new table named DishInventory.

DishInventory table

inventoryID	dishID
1	1
2	2

3NF: Satisfies 3NF, no transitive dependencies

Payment table

paymentID	orderID	paymentMethod	paymentDate
1	1	Card	2024-11-30 00:00:00
2	2	Cash	2024-11-30 00:00:00

1NF: Satisfies 1NF

2NF: Satisfies 2NF, no partial key dependencies

3NF: Satisfies 3NF, no transitive dependencies

V. Creating the Database Schema Using SQL

```
CREATE DATABASE RestaurantDB;
```

```
USE RestaurantDB;
```

```
CREATE TABLE Customer (  
    customerID INT AUTO_INCREMENT PRIMARY KEY,  
    firstName VARCHAR(50) NOT NULL,  
    lastName VARCHAR(50) NOT NULL,  
    phoneNumber VARCHAR(15),  
    address VARCHAR(150),  
    email VARCHAR(100) UNIQUE
```

```
);
```

```
CREATE TABLE Employee (  
    employeeID INT AUTO_INCREMENT PRIMARY KEY,  
    firstName VARCHAR(50) NOT NULL,  
    lastName VARCHAR(50) NOT NULL,  
    phoneNumber VARCHAR(15),  
    address VARCHAR(150),  
    role VARCHAR(50) NOT NULL,  
    email VARCHAR(100) UNIQUE  
    payRatePerHour FLOAT NOT NULL
```

```
);
```

```
CREATE TABLE Order (  
    orderID INT AUTO_INCREMENT PRIMARY KEY,  
    employeeID INT NOT NULL,  
    customerID INT NOT NULL,  
    orderDate DATE NOT NULL,  
    orderType VARCHAR(50) NOT NULL  
    totalPrice FLOAT NOT NULL
```

```
);
```

```
CREATE TABLE DishOrder (  
    orderID INT AUTO_INCREMENT PRIMARY KEY,  
    employeeID VARCHAR(50) NOT NULL,  
    quantity INT NOT NULL
```

```
);
```

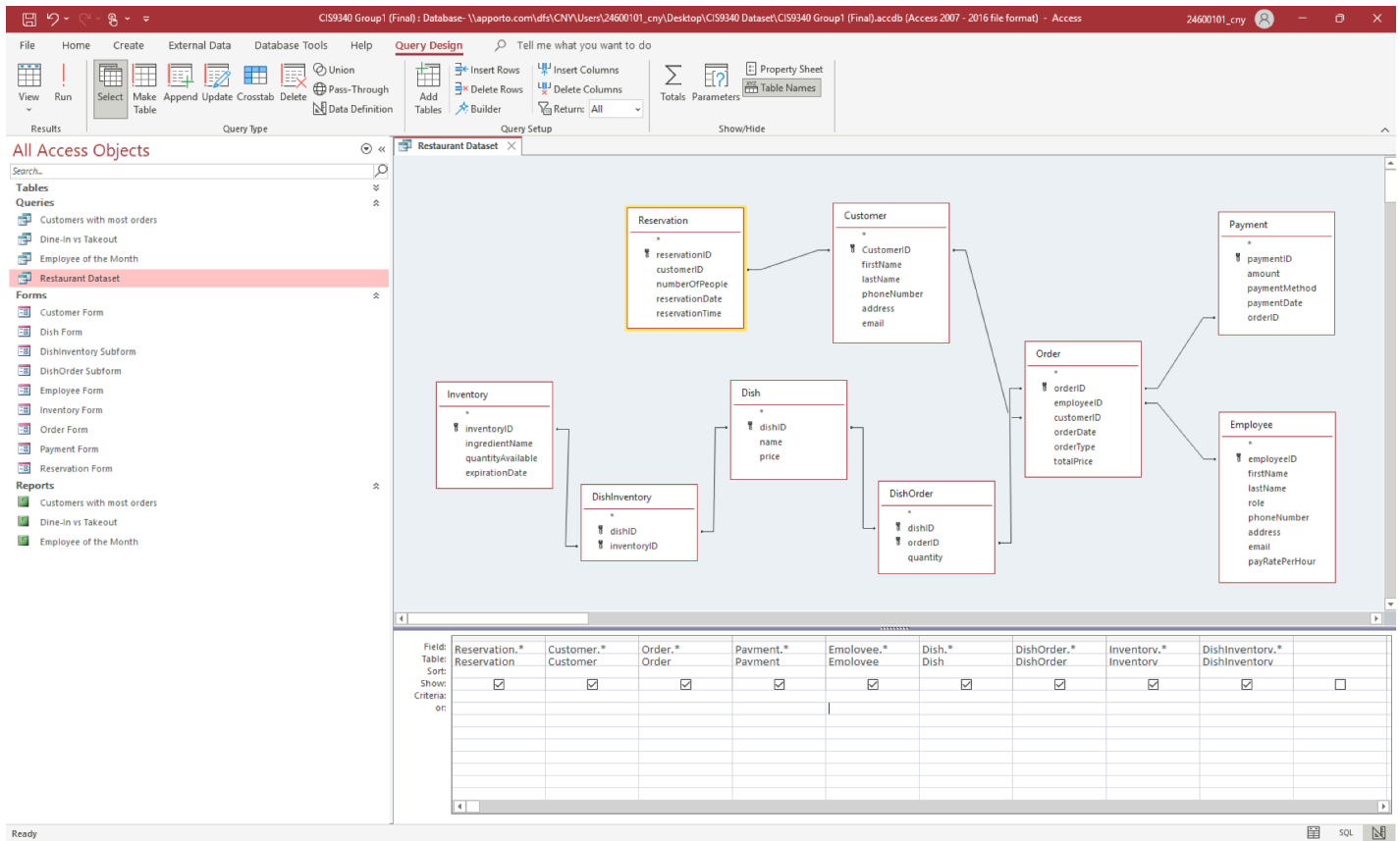
```
CREATE TABLE Dish (  
    dishID INT AUTO_INCREMENT PRIMARY KEY,  
    name VARCHAR(50) NOT NULL,  
    price FLOAT NOT NULL
```

```
);
```



```
CREATE TABLE DishInventory (  
    inventoryID INT NOT NULL,  
    dishID INT NOT NULL  
);  
CREATE TABLE Reservation (  
    reservationID INT AUTO_INCREMENT PRIMARY KEY,  
    customerID INT NOT NULL,  
    numberOfPeople INT NOT NULL,  
    reservationDate DATE NOT NULL  
    reservationTime  
);  
CREATE TABLE Inventory (  
    inventoryID INT AUTO_INCREMENT PRIMARY KEY,  
    ingredientName VARCHAR(50) NOT NULL,  
    quantityAvailable INT NOT NULL  
    expirationDate DATE NOT NULL  
);  
CREATE TABLE Payment (  
    paymentID INT AUTO_INCREMENT PRIMARY KEY,  
    amount FLOAT NOT NULL,  
    paymentMethod VARCHAR(50) NOT NULL  
    paymentDate DATE NOT NULL  
    orderID INT NOT NULL  
);
```

VI. Database Schema creation in Access









SELECT *

FROM (((((Employee INNER JOIN (Reservation INNER JOIN ((Customer INNER JOIN [Order] ON Customer.CustomerID = Order.customerID) INNER JOIN DishOrder ON Order.orderID = DishOrder.orderID) ON Reservation.customerID = Customer.CustomerID) ON Employee.employeeID = Order.employeeID) INNER JOIN Payment ON Order.orderID = Payment.orderID) INNER JOIN Dish ON DishOrder.dishID = Dish.dishID) INNER JOIN DishInventory ON Dish.dishID = DishInventory.dishID) INNER JOIN Inventory ON DishInventory.inventoryID = Inventory.inventoryID;


VII. Database Application

A. Forms:

 Customer Entry

Customer ID	<input type="text" value="1"/>	New Customer		
First Name	<input type="text" value="Kathryn"/>			
Last Name	<input type="text" value="Jacobson"/>			
Phone Number	<input type="text" value="(744) 577-6982"/>			
Address	<input type="text" value="47652 Chesley Knolls"/>			
E-mail	<input type="text" value="Kathryn.Jacobson@hotmail.com"/>		Send E-mail	Close

Customer Management: Maintain a detailed database of our customers, including their contact information and purchase history. This helps us provide personalized service, run targeted marketing campaigns, and build customer loyalty.

 Dishes

Dish ID	<input type="text" value="1"/>
Name	<input type="text" value="Pork Belly Buns"/>
Price	<input type="text" value="5"/>

Inventory


▶ Ingredient

▼

Add Inventory

Delete Inventory


Record: 1 of 1

 No Filter

Search

Save

Menu/Dishes Management: Stay on top of food dishes by tracking their pricing and popularity, and don't hesitate to adjust prices when needed for optimal success.

 Employee Directory

List of Employees

Add Employee

Remove Employee

Save

Employee ID

1

First Name

Triston

Last Name


Boyer

Role

Manager

Phone Number

(387) 304-7358



Address

3167 Castle Road

E-mail

Triston_Boyer@yahoo.com

Current Pay/hr


28

Edit Pay

Save Pay

Close

Employee Directory: Maintain a comprehensive and accurate directory encompassing staff roles, contact information, and salary rates.

 Kitchen Inventory

List of Ingredients

Add Ingredients

Remove Ingredients

Inventory ID


1

Dish ID

20

Ingredient Name

jicama




Quantity Available

32

Expiration Date

2025-11-12 0:00:00



Save

Close

Kitchen Inventory: Effectively manage inventory by recording inventory levels, monitoring expiration dates, and minimizing waste. Maintain a consistent supply of essential ingredients at all times.



Order Information

Order ID	<input type="text"/>	<input type="button" value="Add Order"/>	<input type="button" value="Remove Order"/>						
Employee	<input type="text" value="Lyric Hirthe"/>								
Customer	<input type="text" value="Brittany Towne"/>								
Order Date	<input type="text" value="2024-02-18 0:00:00"/>								
Order Type	<input type="text" value="Dine-In"/>								
Dishes	<table><tr><td>Dish</td><td><input type="text" value="Nashi Pear And Pear Tart"/></td><td><input type="button" value="Add Dish"/></td></tr><tr><td>Quantity</td><td><input type="text" value="1"/></td><td><input type="button" value="Delete Dish"/></td></tr></table>			Dish	<input type="text" value="Nashi Pear And Pear Tart"/>	<input type="button" value="Add Dish"/>	Quantity	<input type="text" value="1"/>	<input type="button" value="Delete Dish"/>
Dish	<input type="text" value="Nashi Pear And Pear Tart"/>	<input type="button" value="Add Dish"/>							
Quantity	<input type="text" value="1"/>	<input type="button" value="Delete Dish"/>							
<div>Record: 1 of 2 <input type="button" value="Previous"/> <input type="button" value="Next"/> <input type="button" value="Refresh"/> No Filter <input type="text" value="Search"/></div>									
Total Price	<input type="text" value="112"/>								

Order Management: Ensure the accurate and timely processing of orders. Produce comprehensive reports regarding sales and revenue.

Payment Information			
Payment ID	<input type="text" value="1"/>	<button>Change ID</button>	
Amount	<input type="text" value="112"/>	<button>Edit Amount</button>	
Payment Method	<input type="text" value="Card"/>	<button>Change Payment Method</button>	
Payment Date	<input type="text" value="2024-11-30 0:00:00"/>	<button>Change Date</button>	
Order ID	<input type="text" value="1"/>	<button>Change Order</button>	<div><button>Save</button><button>Quit</button></div>

Payment Processing: Simplify payment processing, accept various payment methods, and generate accurate reports on daily revenue.

Reservation's			
Reservation ID	<input type="text" value=""/>	<button>Add Reservation</button>	<button>Cancel Reservation</button>
Customer ID	<input type="text" value="1"/>	<button>Add Customer</button>	<button>Edit Customer Information</button>
Number of Guests	<input type="text" value="2"/>	<button>Add Guest(s)</button>	<button>Remove Guest(s)</button>
Reservation Date	<input type="text" value="2025-06-27 0:00:00"/>	<button>Change Reservation Date</button>	
Reservation Time	<input type="text" value="03:16:00"/>	<button>Save</button>	

Reservation Management: Easily manage reservations, optimize table utilization, and reduce wait time, ensuring a seamless dining experience for customers.

DishOrder Subform

Dish
Add Dish

Quantity
Delete Dish

Dish Order: Designed to streamline the process of taking and managing customer orders.

DishInventory Subform

Ingredient
Add Inventory

Delete Inventory

Dish Inventory: Tracks all the ingredients required to prepare dishes. Allows staff to update inventory in real-time as ingredients are used.

B. Quarries & Reports:

Customers with most orders			
			Sunday, December 8, 2024
			2:09:00 PM
firstName	lastName	phoneNumber	Number Of Orders
Broderick	Trantow	(817) 427-5062	6
Jefferey	Baumbach	(611) 817-6819	5
Jodie	Cormier	(377) 264-4504	4
Hilma	Sawayn	(277) 841-6020	4
Erna	Morissette	(219) 285-6514	4
Breanne	Kutch	(640) 730-2973	4
Aleen	Wisozk	(978) 537-7887	4
Kathryn	Jacobson	(744) 577-6982	3
Garnet	Yost	(235) 906-3425	3
Brittany	Towne	(723) 661-2235	3

This quarry shows customers with the most orders at the restaurant. The quarry can be modified to specify a time frame to see results monthly or yearly. This can be useful to management if they decide to implement a rewards program in the future.

```
SQL: SELECT DISTINCTROW Customer.firstName, Customer.lastName,
Customer.phoneNumber, Count(*) AS [Count Of Order]
FROM Customer INNER JOIN [Order] ON Customer.[CustomerID] = Order.[customerID] GROUP
BY Customer.firstName, Customer.lastName, Customer.phoneNumber;
```

Comparison of Dine-In vs Takeout			Sunday, December 8, 2024
			3:10:51 PM
Order Type	Total \$ Ammount	Number of Orders	
Dine-In	1533	23	
Takeout	1766	17	

This quarry shows the comparison of dine-in and takeout. For example which mode of order generated the most revenue and which mode received the most orders. This is important information for management to consider when making future plans.

```
SQL: SELECT DISTINCTROW Order.orderType, Sum(Order.totalPrice) AS [Sum Of
totalPrice], Count(*) AS [Count Of Order] FROM [Order]
GROUP BY Order.orderType;
```

Employee of the Month				Sunday, December 8, 2024
				3:33:09 PM
First Name	Last Name	Pay Rate Per Hour	Total Sales Generated	
Maximillian	Parisian	20	918	
Cicero	Kling-Ferry	26	862	
Triston	Boyer	28	826	
Lyric	Hirthe	10	693	

This quarry shows which employee generated the most revenue. Its something to consider if management would like to implement an employee of the month system.

```
SQL: SELECT Employee.firstName, Employee.lastName, Employee.payRatePerHour,
SUM(Order.totalPrice) AS [Sum Of totalPrice]
FROM Employee INNER JOIN [Order] ON Employee.[employeeID] = Order.[employeeID]
GROUP BY Employee.firstName, Employee.lastName, Employee.payRatePerHour
ORDER BY SUM(Order.totalPrice) DESC;
```


VIII. Conclusion

Our database system effectively integrates multiple important operations, guaranteeing a seamless workflow and improving efficiency. This system enables comprehensive customer management, allowing the restaurant to offer personalized services and enhance customer satisfaction. The process of normalization, combined with SQL implementation, plays a vital role in maintaining data consistency and integrity while also facilitating easier maintenance of databases. Overall, our database system serves as a powerful tool for streamlining operations and providing an excellent dining experience for customers while supporting the restaurant's long-term success.