**Ma's Tacos Restaurant Data System**

**Table of Contents**

[**1.** Project Definition 8](#_Toc198047757)

[1.1 Background Statement 8](#_Toc198047758)

[1.2 Intentions 8](#_Toc198047759)

[1.3 Values 8](#_Toc198047760)

[1.4 Focus 8](#_Toc198047761)

[1.4.1 Breadth 8](#_Toc198047762)

[Processes: 8](#_Toc198047763)

[Information: 9](#_Toc198047764)

[Events: 9](#_Toc198047765)

[Groups/Org Units: 9](#_Toc198047766)

[Location: 9](#_Toc198047767)

[1.4.2 Depth 9](#_Toc198047768)

[1.4.3 Emphasized Perspectives 9](#_Toc198047769)

[1.4.4 Universality 10](#_Toc198047770)

[2. Artifacts and Collaboration 11](#_Toc198047771)

[2.1 Customer Engagement System 11](#_Toc198047772)

[2.2 Reservation System 11](#_Toc198047773)

[2.3 Menu Item Tracking 11](#_Toc198047774)

[2.4 Survey and Feedback System 11](#_Toc198047775)

[2.5 Analytics Dashboard 12](#_Toc198047776)

[3. Conceptual Data Model Fragment 13](#_Toc198047777)

[3.1 Entity Relationship Diagram (Conceptual Level) 13](#_Toc198047778)

[CUSTOMER 13](#_Toc198047779)

[RESERVATION 13](#_Toc198047780)

[TIME\_SLOT 13](#_Toc198047781)

[TABLE 14](#_Toc198047782)

[ORDER 14](#_Toc198047783)

[ORDER\_ITEM 14](#_Toc198047784)

[MENU\_ITEM 14](#_Toc198047785)

[FEEDBACK 14](#_Toc198047786)

[4. Logical Data Model Fragment 15](#_Toc198047787)

[4.1 Entity Relationship Diagram (Logical Level) 15](#_Toc198047788)

[4.2 Logical Relationships 15](#_Toc198047789)

[Customer to Reservation (1:N) 15](#_Toc198047790)

[Reservation to TimeSlot (N:1) 15](#_Toc198047791)

[Reservation to Table (N:1) 15](#_Toc198047792)

[Customer to Order (1:N) 15](#_Toc198047793)

[Order to OrderItem (1:N) 16](#_Toc198047794)

[MenuItem to OrderItem (1:N) 16](#_Toc198047795)

[MenuItem to MenuCategory (N:1) 16](#_Toc198047796)

[Order to Feedback (1:1) 16](#_Toc198047797)

[Customer to Feedback (1:N) 16](#_Toc198047798)

[Reservation to Order (1:1) 16](#_Toc198047799)

[5. Physical Data Model Fragment 17](#_Toc198047800)

[5.1 Database Schema (MySQL) 17](#_Toc198047801)

[5.2 Physical Data Model Considerations 17](#_Toc198047802)

[**Indexes** have been added to optimize common query patterns: 17](#_Toc198047803)

[**Data Types** have been chosen to balance storage efficiency and performance: 17](#_Toc198047804)

[**Constraints** implemented include: 18](#_Toc198047805)

[Performance Considerations: 18](#_Toc198047806)

[6. DDL & DML SQL Scripts 19](#_Toc198047807)

[6.1 DDL (Data Definition Language) Scripts 19](#_Toc198047808)

[-- Customer Table 19](#_Toc198047809)

[-- Menu Item Table 20](#_Toc198047810)

[-- Time Slot Table 20](#_Toc198047811)

[-- Table Table (using backticks because Table is a reserved word) 21](#_Toc198047812)

[-- Reservation Table 21](#_Toc198047813)

[-- Order Table (using backticks because Order is a reserved word) 22](#_Toc198047814)

[-- Order Item Table 22](#_Toc198047815)

[-- Feedback Table 23](#_Toc198047816)

[6.2 DML (Data Manipulation Language) Scripts 23](#_Toc198047817)

[-- Sample data for MenuCategory 23](#_Toc198047818)

[-- Sample data for MenuItem (abbreviated for clarity) 24](#_Toc198047819)

[-- Sample data for Table 24](#_Toc198047820)

[-- Sample data for TimeSlot (abbreviated for clarity) 25](#_Toc198047821)

[-- Sample data for Customer 25](#_Toc198047822)

[-- Stored Procedure for Order Processing 25](#_Toc198047823)

[-- Create the order 26](#_Toc198047824)

[-- Get the order ID 26](#_Toc198047825)

[-- Return the order ID 26](#_Toc198047826)

[-- Get the unit price from the menu item 26](#_Toc198047827)

[-- Calculate the subtotal 27](#_Toc198047828)

[-- Insert the order item 27](#_Toc198047829)

[-- Update the order total 27](#_Toc198047830)

[-- Update the order status 27](#_Toc198047831)

[-- Create a feedback record 27](#_Toc198047832)

[-- Add loyalty points to the customer 28](#_Toc198047833)

[6.4 Sample Business Analytics Queries 28](#_Toc198047834)

[-- 1. Popular Menu Items by Revenue 28](#_Toc198047835)

[-- 2. Customer Loyalty Program Analysis 29](#_Toc198047836)

[7. Analysis and Design Artifacts 31](#_Toc198047837)

[7.1 Individual Project Definition 31](#_Toc198047838)

[Customer Engagement Management 31](#_Toc198047839)

[Order Processing and Analysis 31](#_Toc198047840)

[Business Intelligence Components 31](#_Toc198047841)

[7.2 Data Flow Diagram 32](#_Toc198047842)

[Key Data Flow Processes: 32](#_Toc198047843)

[Customer Information Flow 32](#_Toc198047844)

[Order Processing Flow 33](#_Toc198047845)

[Reservation Management Flow 33](#_Toc198047846)

[Analytics Data Flow 33](#_Toc198047847)

[7.3.1 Reservation State Transitions 33](#_Toc198047848)

[States: 33](#_Toc198047849)

[Transitions: 34](#_Toc198047850)

[7.3.2 Order State Transitions 34](#_Toc198047851)

[States: 34](#_Toc198047852)

[Transitions: 34](#_Toc198047853)

[7.3.3 Feedback Collection State Transitions 34](#_Toc198047854)

[States: 34](#_Toc198047855)

[Transitions: 35](#_Toc198047856)

[7.4 Use Case Diagram 35](#_Toc198047857)

[Primary Actors: 35](#_Toc198047858)

[Key Use Cases: 35](#_Toc198047859)

[Architecture Diagram: 36](#_Toc198047860)

[8. Naming Conventions and Standards 37](#_Toc198047861)

[8.1 Database Object Naming Conventions 37](#_Toc198047862)

[8.1.1 Tables 37](#_Toc198047863)

[8.1.2 Columns 37](#_Toc198047864)

[8.1.3 Constraints and Indexes 38](#_Toc198047865)

[8.2 SQL Coding Standards 38](#_Toc198047866)

[8.2.1 SQL Keywords and Formatting 38](#_Toc198047867)

[8.2.2 Query Structure Example 38](#_Toc198047868)

[8.2.3 Stored Procedures 39](#_Toc198047869)

[8.2.4 Triggers 41](#_Toc198047870)

[8.2.5 Views 41](#_Toc198047871)

[8.3 Data Modeling Standards 41](#_Toc198047872)

[8.3.1 Entity Relationship Diagram 41](#_Toc198047873)

[8.3.2 Data Types 41](#_Toc198047874)

[8.3.3 Normalization 42](#_Toc198047875)

[8.4 Documentation Standards 42](#_Toc198047876)

[8.4.1 Data Dictionary 42](#_Toc198047877)

[8.4.2 Model Versioning 44](#_Toc198047878)

[8.4.3 Security Classification 44](#_Toc198047879)

[8.5 Implementation Standards 45](#_Toc198047880)

[8.5.1 Database Creation 45](#_Toc198047881)

[8.5.2 Performance Considerations 46](#_Toc198047882)

[9. Conclusion 47](#_Toc198047883)

# **1.** Project Definition

## 1.1 Background Statement

Ma's Tacos is a family-owned Mexican restaurant seeking to modernize its operations through data-driven solutions. Our consulting company has been engaged to provide comprehensive data technology solutions that will enhance customer experience, optimize operations, and drive business growth.

## 1.2 Intentions

We aim to implement data technology solutions that will:

* Ensure seamless monetary transactions
* Provide payment type analysis and insights
* Analyze customer engagement data from food orders and surveys
* Support effective marketing strategies to increase restaurant traffic
* Establish a loyalty program to encourage repeat business

## 1.3 Values

* Accuracy, relevance, and meaningfulness of data
* Elimination of redundant or unnecessary statistics
* Security as a fundamental aspect of all solutions

## 1.4 Focus

The focus of our data management system for Ma's Tacos is clearly defined to ensure targeted and effective implementation.

## 1.4.1 Breadth

Our effort encompasses customer experience management through:

## Processes:

* Reservations system
* Marketing strategies
* Customer surveys system

## Information:

* Financial data
* Customer data
* Employee data
* Inventory process

## Events:

* Transaction events
* Inventory events
* Scheduling events

## Groups/Org Units:

* Customers
* Employees
* Food/Inventory
* Payroll
* Financial analysts

## Location:

* Minnesota

## 1.4.2 Depth

Our solution requires detailed and precise data to support informed strategic business decisions.

## 1.4.3 Emphasized Perspectives

* Customer engagement metrics
* Analytics based on customer interaction
* Inventory management in relation to customer preferences

## 1.4.4 Universality

Our solution is primarily designed for Ma's Tacos with potential to be adapted for other restaurants in the future.

# 2. Artifacts and Collaboration

## 2.1 Customer Engagement System

**Purpose:** Tracking customer interactions, preferences, and loyalty **Collaborators:** Restaurant management, marketing team, development team **Agreements Needed:**

* Definition of loyalty points accrual rates and redemption values
* Customer data fields to be collected
* Privacy and data retention policies
* Marketing communication opt-in requirements

## 2.2 Reservation System

**Purpose:** Managing table allocation and customer time slots **Collaborators:** Restaurant hosts, management, customers, development team **Agreements Needed:**

* Time slot definitions
* Party size limitations
* Special requests handling
* Cancellation policies
* No-show management

## 2.3 Menu Item Tracking

**Purpose:** Analyzing popular items and inventory needs **Collaborators:** Kitchen staff, management, procurement team **Agreements Needed:**

* Menu categorization system
* Popularity metrics definition
* Inventory thresholds
* Seasonal menu item rotation plan

## 2.4 Survey and Feedback System

**Purpose:** Collecting and analyzing customer opinions **Collaborators:** Customers, management, staff trainers **Agreements Needed:**

* Survey question standardization
* Rating scale definitions
* Follow-up triggers for negative feedback
* Performance metric definitions based on feedback

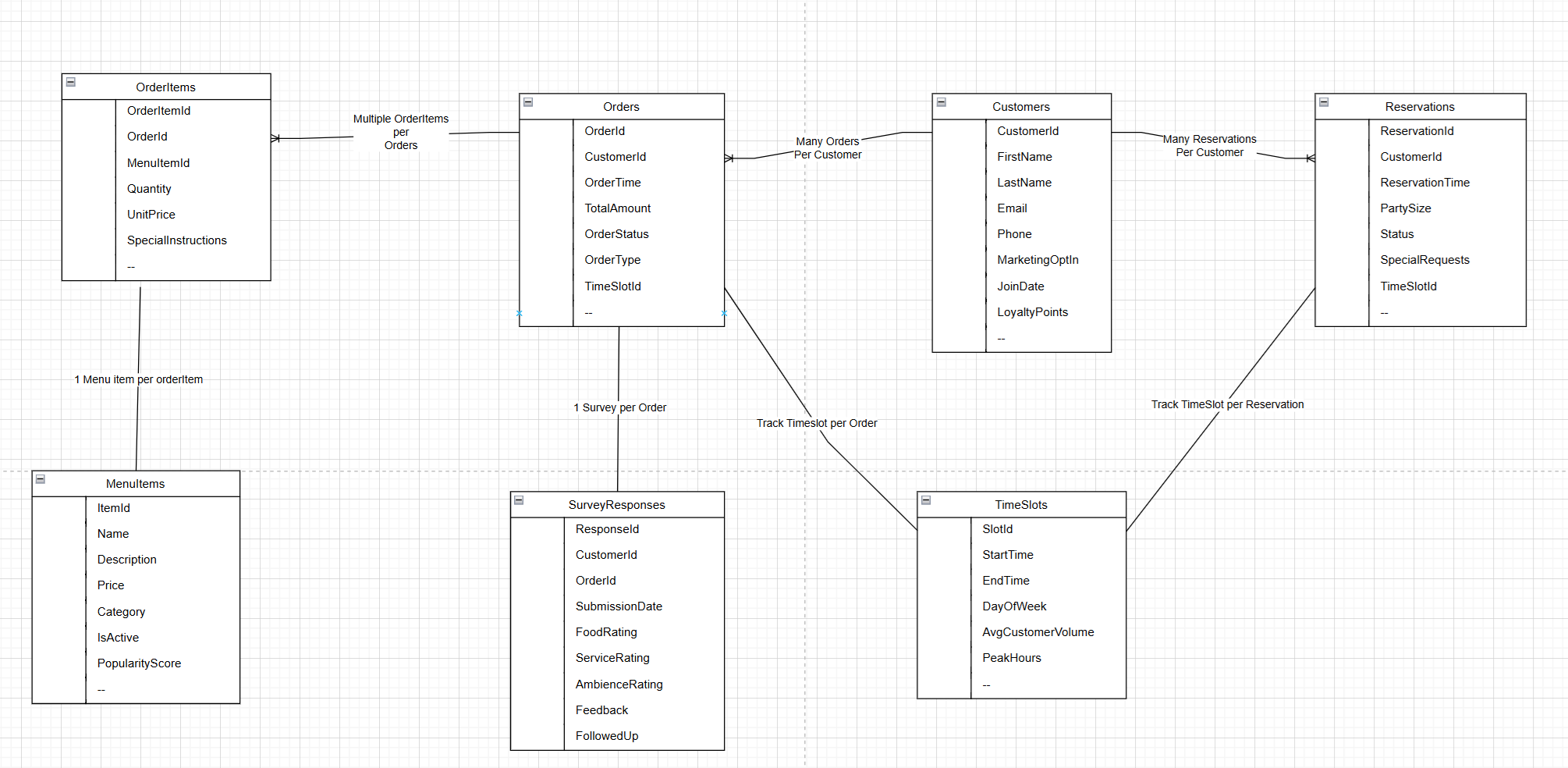
## 2.5 Analytics Dashboard

**Purpose:** Visualizing key business metrics for decision-making **Collaborators:** Management, financial analysts, development team **Agreements Needed:**

* Key performance indicators
* Data refresh frequency
* User access levels
* Alert thresholds

# 3. Conceptual Data Model Fragment

## 3.1 Entity Relationship Diagram (Conceptual Level)

3.2 Entity Descriptions

### CUSTOMER

* Conceptual attributes: CustomerID, Name, Contact Information, Loyalty Points
* Description: Represents individuals who patronize the restaurant
* Relationships: Makes reservations, places orders, provides feedback

### RESERVATION

* Conceptual attributes: ReservationID, Date, Party Size, Status
* Description: Records booking of tables for specific time slots
* Relationships: Made by customers, allocated to time slots, assigned to tables

### TIME\_SLOT

* Conceptual attributes: TimeSlotID, Start Time, End Time, Availability
* Description: Defines specific periods when tables can be reserved
* Relationships: Contains reservations, associated with tables

### TABLE

* Conceptual attributes: TableID, Capacity, Location
* Description: Physical dining surfaces in the restaurant
* Relationships: Assigned to reservations during time slots

### ORDER

* Conceptual attributes: OrderID, Date/Time, Total Amount, Status
* Description: Records transactions of food/beverage purchases
* Relationships: Placed by customers, contains order items

### ORDER\_ITEM

* Conceptual attributes: OrderItemID, Quantity, Subtotal
* Description: Individual items within an order
* Relationships: Belongs to an order, refers to menu items

### MENU\_ITEM

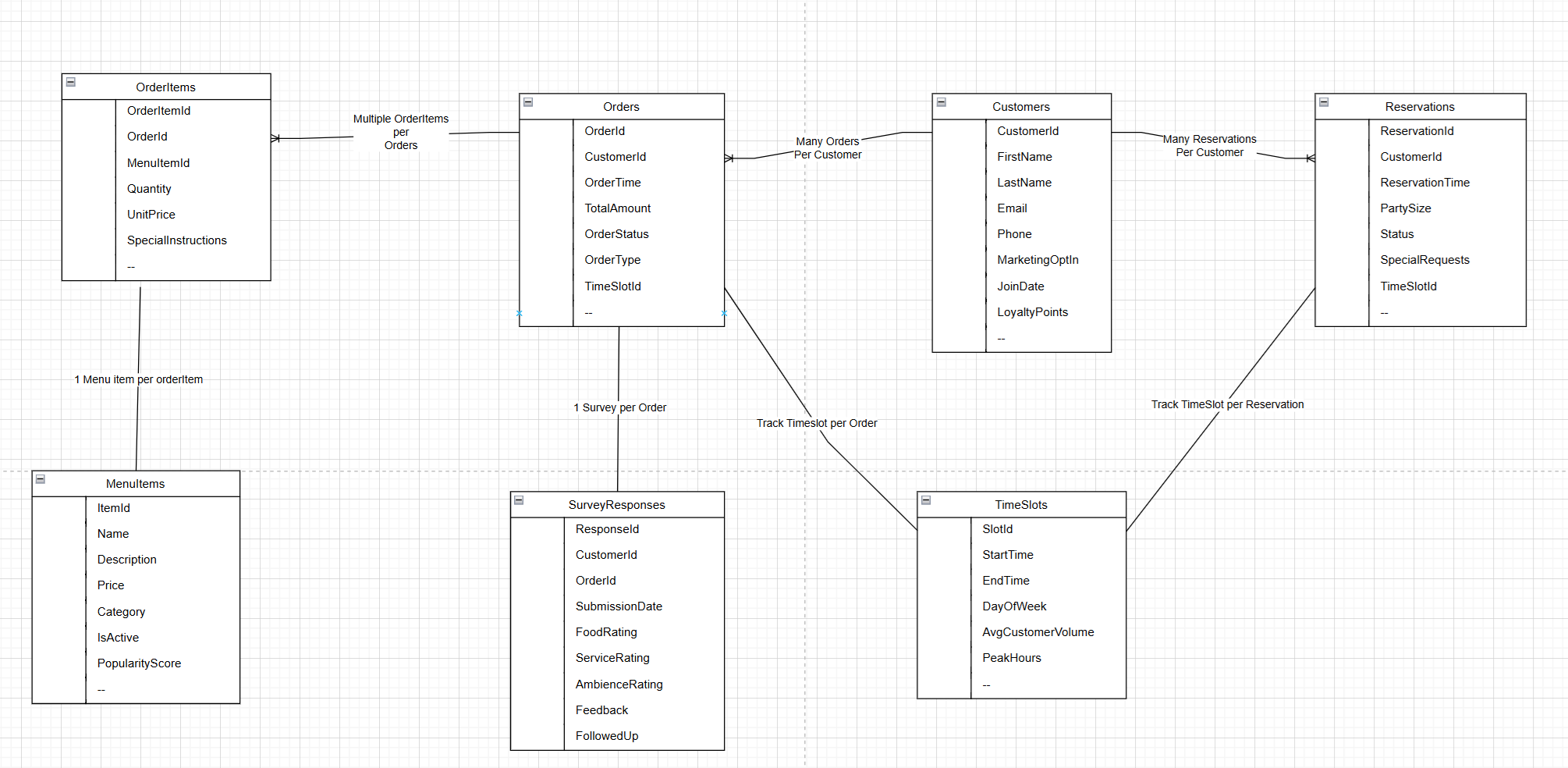
* Conceptual attributes: MenuItemID, Name, Description, Price, Category
* Description: Food and beverage offerings available for purchase
* Relationships: Included in order items

### FEEDBACK

* Conceptual attributes: FeedbackID, Rating, Comments, Date
* Description: Customer opinions and ratings about their experience
* Relationships: Provided by customers, associated with orders

# 4. Logical Data Model Fragment

## 4.1 Entity Relationship Diagram (Logical Level)

****

## 4.2 Logical Relationships

### Customer to Reservation (1:N)

* + A customer can make multiple reservations
  + Each reservation belongs to exactly one customer

### Reservation to TimeSlot (N:1)

* + Multiple reservations can be assigned to a time slot (within capacity limits)
  + Each reservation belongs to exactly one time slot

### Reservation to Table (N:1)

* + Multiple reservations can be assigned to a table (at different times)
  + Each reservation is assigned to exactly one table

### Customer to Order (1:N)

* + A customer can place multiple orders
  + Each order belongs to exactly one customer

### Order to OrderItem (1:N)

* + An order contains multiple order items
  + Each order item belongs to exactly one order

### MenuItem to OrderItem (1:N)

* + A menu item can appear in multiple order items
  + Each order item references exactly one menu item

### MenuItem to MenuCategory (N:1)

* + Multiple menu items can belong to a category
  + Each menu item belongs to exactly one category

### Order to Feedback (1:1)

* + An order can have one feedback record
  + Each feedback record is associated with exactly one order

### Customer to Feedback (1:N)

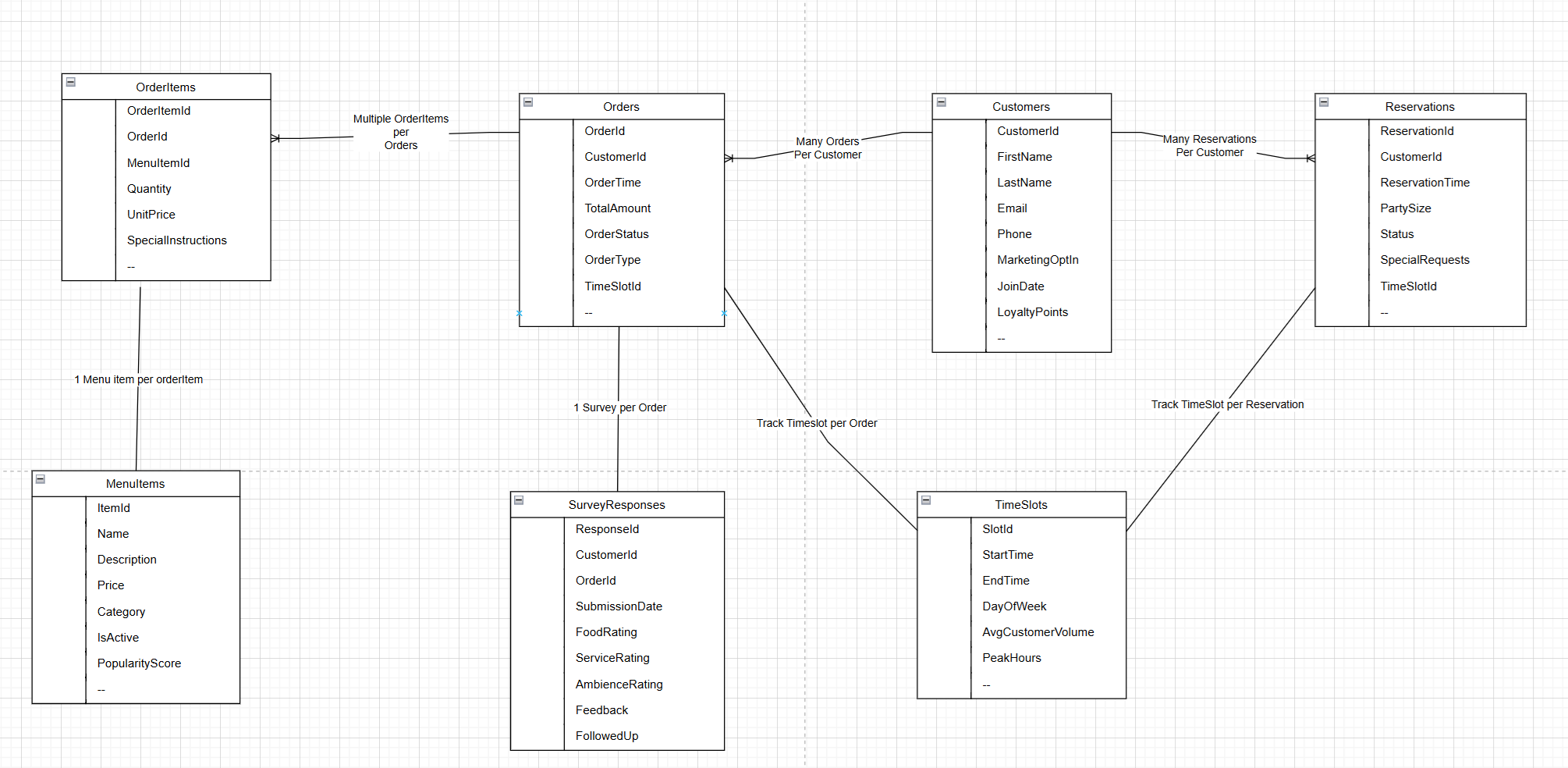
* + A customer can provide multiple feedback records
  + Each feedback record is from exactly one customer

### Reservation to Order (1:1)

* + A reservation can be associated with one order
  + An order may or may not be associated with a reservation

# 5. Physical Data Model Fragment

## 5.1 Database Schema (MySQL)

****

## 5.2 Physical Data Model Considerations

### **Indexes** have been added to optimize common query patterns:

* + Customer lookups by email and name
  + Menu items filtering by category and active status
  + Time slot searches by day and time
  + Table filtering by capacity
  + Reservation filtering by date and status
  + Order filtering by customer, date, and status
  + Order item lookups by order
  + Feedback analysis by ratings

### **Data Types** have been chosen to balance storage efficiency and performance:

* + VARCHAR for variable-length strings
  + TEXT for longer text fields
  + DECIMAL for monetary values to ensure precision
  + TINYINT(1) for boolean values
  + ENUM for fields with a fixed set of values

### **Constraints** implemented include:

* + Primary keys on all tables
  + Foreign key relationships with appropriate references
  + Unique constraints where applicable
  + Default values for common fields
  + NOT NULL constraints for required fields

### Performance Considerations:

* + Appropriate use of indexes to speed up common queries
  + ENGINE=InnoDB for transaction support and referential integrity
  + Careful selection of data types to minimize storage requirements

# 6. DDL & DML SQL Scripts

## 6.1 DDL (Data Definition Language) Scripts

-- Database Creation

CREATE DATABASE IF NOT EXISTS MasTacos;

USE MasTacos;

### -- Customer Table

CREATE TABLE Customer (

CustomerID INT AUTO\_INCREMENT PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Email VARCHAR(100) UNIQUE,

Phone VARCHAR(20),

LoyaltyPoints INT DEFAULT 0,

OptInMarketing TINYINT(1) DEFAULT 0,

CreateDate DATETIME DEFAULT CURRENT\_TIMESTAMP,

INDEX idx\_customer\_email (Email),

INDEX idx\_customer\_name (LastName, FirstName)

) ENGINE=InnoDB;

-- Menu Category Table

CREATE TABLE MenuCategory (

CategoryID INT AUTO\_INCREMENT PRIMARY KEY,

CategoryName VARCHAR(50) NOT NULL UNIQUE,

Description VARCHAR(255)

) ENGINE=InnoDB;

### -- Menu Item Table

CREATE TABLE MenuItem (

MenuItemID INT AUTO\_INCREMENT PRIMARY KEY,

CategoryID INT NOT NULL,

ItemName VARCHAR(100) NOT NULL,

Description TEXT,

Price DECIMAL(10,2) NOT NULL,

IsActive TINYINT(1) DEFAULT 1,

IsSpecial TINYINT(1) DEFAULT 0,

FOREIGN KEY (CategoryID) REFERENCES MenuCategory(CategoryID),

INDEX idx\_menu\_category (CategoryID),

INDEX idx\_menu\_active (IsActive)

) ENGINE=InnoDB;

### -- Time Slot Table

CREATE TABLE TimeSlot (

TimeSlotID INT AUTO\_INCREMENT PRIMARY KEY,

StartTime TIME NOT NULL,

EndTime TIME NOT NULL,

DayOfWeek TINYINT NOT NULL, -- 1 = Monday, 7 = Sunday

MaxCapacity INT NOT NULL,

INDEX idx\_timeslot\_day\_time (DayOfWeek, StartTime)

) ENGINE=InnoDB;

### -- Table Table (using backticks because Table is a reserved word)

CREATE TABLE `Table` (

TableID INT AUTO\_INCREMENT PRIMARY KEY,

TableNumber VARCHAR(10) NOT NULL UNIQUE,

Capacity INT NOT NULL,

Location ENUM('Indoor', 'Outdoor', 'Bar', 'Private') NOT NULL,

INDEX idx\_table\_capacity (Capacity)

) ENGINE=InnoDB;

### -- Reservation Table

CREATE TABLE Reservation (

ReservationID INT AUTO\_INCREMENT PRIMARY KEY,

CustomerID INT NOT NULL,

TimeSlotID INT NOT NULL,

TableID INT NOT NULL,

ReservationDate DATE NOT NULL,

PartySize INT NOT NULL,

SpecialRequests TEXT,

Status ENUM('Pending', 'Confirmed', 'Seated', 'Completed', 'Cancelled') DEFAULT 'Pending',

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),

FOREIGN KEY (TimeSlotID) REFERENCES TimeSlot(TimeSlotID),

FOREIGN KEY (TableID) REFERENCES `Table`(TableID),

INDEX idx\_reservation\_date (ReservationDate),

INDEX idx\_reservation\_status (Status)

) ENGINE=InnoDB;

### -- Order Table (using backticks because Order is a reserved word)

CREATE TABLE `Order` (

OrderID INT AUTO\_INCREMENT PRIMARY KEY,

CustomerID INT NOT NULL,

ReservationID INT,

OrderDate DATETIME DEFAULT CURRENT\_TIMESTAMP,

TotalAmount DECIMAL(10,2) NOT NULL DEFAULT 0.00,

PaymentType ENUM('Cash', 'Credit', 'Debit', 'Gift Card'),

Status ENUM('New', 'InProgress', 'Ready', 'Served', 'Completed') DEFAULT 'New',

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),

FOREIGN KEY (ReservationID) REFERENCES Reservation(ReservationID),

INDEX idx\_order\_customer (CustomerID),

INDEX idx\_order\_date (OrderDate),

INDEX idx\_order\_status (Status)

) ENGINE=InnoDB;

### -- Order Item Table

CREATE TABLE OrderItem (

OrderItemID INT AUTO\_INCREMENT PRIMARY KEY,

OrderID INT NOT NULL,

MenuItemID INT NOT NULL,

Quantity INT NOT NULL DEFAULT 1,

UnitPrice DECIMAL(10,2) NOT NULL,

Subtotal DECIMAL(10,2) NOT NULL,

SpecialInstructions TEXT,

FOREIGN KEY (OrderID) REFERENCES `Order`(OrderID) ON DELETE CASCADE,

FOREIGN KEY (MenuItemID) REFERENCES MenuItem(MenuItemID),

INDEX idx\_orderitem\_order (OrderID)

) ENGINE=InnoDB;

### -- Feedback Table

CREATE TABLE Feedback (

FeedbackID INT AUTO\_INCREMENT PRIMARY KEY,

OrderID INT NOT NULL,

CustomerID INT NOT NULL,

FoodRating TINYINT,

ServiceRating TINYINT,

AmbienceRating TINYINT,

Comments TEXT,

SubmissionDate DATETIME DEFAULT CURRENT\_TIMESTAMP,

FollowUpRequired TINYINT(1) DEFAULT 0,

FollowUpCompleted TINYINT(1) DEFAULT 0,

FOREIGN KEY (OrderID) REFERENCES `Order`(OrderID),

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),

UNIQUE KEY (OrderID),

INDEX idx\_feedback\_ratings (FoodRating, ServiceRating)

) ENGINE=InnoDB;

## 6.2 DML (Data Manipulation Language) Scripts

### -- Sample data for MenuCategory

INSERT INTO MenuCategory (CategoryName, Description) VALUES

('Appetizers', 'Starters and small plates'),

('Tacos', 'Traditional and specialty tacos'),

('Burritos', 'Filled flour tortilla wraps'),

('Enchiladas', 'Tortillas with filling and sauce'),

('Sides', 'Additional side dishes'),

('Desserts', 'Sweet treats to finish your meal'),

('Beverages', 'Drinks and refreshments');

### -- Sample data for MenuItem (abbreviated for clarity)

INSERT INTO MenuItem (CategoryID, ItemName, Description, Price, IsActive, IsSpecial) VALUES

(1, 'Guacamole & Chips', 'Fresh avocado dip with homemade tortilla chips', 8.99, 1, 0),

(1, 'Queso Fundido', 'Melted cheese with chorizo and peppers', 7.99, 1, 0),

(2, 'Carne Asada Taco', 'Grilled steak with cilantro and onions', 4.50, 1, 0),

(2, 'Pollo Taco', 'Seasoned chicken with lettuce and cheese', 3.99, 1, 0),

(3, 'Carne Asada Burrito', 'Grilled steak with rice, beans, and cheese', 11.99, 1, 0),

(6, 'Churros', 'Fried dough pastry with cinnamon sugar and chocolate sauce', 5.99, 1, 0),

(7, 'Horchata', 'Sweet rice milk with cinnamon', 2.99, 1, 0);

### -- Sample data for Table

INSERT INTO `Table` (TableNumber, Capacity, Location) VALUES

('A1', 2, 'Indoor'),

('A2', 2, 'Indoor'),

('B1', 4, 'Indoor'),

('C1', 6, 'Indoor'),

('D1', 8, 'Private'),

('O1', 4, 'Outdoor'),

('BAR1', 1, 'Bar');

### -- Sample data for TimeSlot (abbreviated for clarity)

INSERT INTO TimeSlot (StartTime, EndTime, DayOfWeek, MaxCapacity) VALUES

('11:00:00', '12:30:00', 1, 30), -- Monday lunch

('17:00:00', '18:30:00', 1, 30), -- Monday dinner

('11:00:00', '12:30:00', 5, 30), -- Friday lunch

('18:30:00', '20:00:00', 5, 35); -- Friday dinner

### -- Sample data for Customer

INSERT INTO Customer (FirstName, LastName, Email, Phone, LoyaltyPoints, OptInMarketing) VALUES

('John', 'Smith', 'john.smith@example.com', '612-555-1234', 150, 1),

('Maria', 'Garcia', 'maria.garcia@example.com', '612-555-2345', 75, 1),

('David', 'Johnson', 'david.johnson@example.com', '612-555-3456', 200, 1);

**6.3 Stored Procedures**

DELIMITER //

### -- Stored Procedure for Order Processing

CREATE PROCEDURE CreateOrder(

IN p\_CustomerID INT,

IN p\_ReservationID INT,

IN p\_PaymentType VARCHAR(20)

)

BEGIN

DECLARE v\_OrderID INT;

### -- Create the order

INSERT INTO `Order` (CustomerID, ReservationID, PaymentType, Status)

VALUES (p\_CustomerID, p\_ReservationID, p\_PaymentType, 'New');

### -- Get the order ID

SET v\_OrderID = LAST\_INSERT\_ID();

### -- Return the order ID

SELECT v\_OrderID AS OrderID;

END //

CREATE PROCEDURE AddOrderItem(

IN p\_OrderID INT,

IN p\_MenuItemID INT,

IN p\_Quantity INT,

IN p\_SpecialInstructions TEXT

)

BEGIN

DECLARE v\_UnitPrice DECIMAL(10,2);

DECLARE v\_Subtotal DECIMAL(10,2);

### -- Get the unit price from the menu item

SELECT Price INTO v\_UnitPrice FROM MenuItem WHERE MenuItemID = p\_MenuItemID;

### -- Calculate the subtotal

SET v\_Subtotal = v\_UnitPrice \* p\_Quantity;

### -- Insert the order item

INSERT INTO OrderItem (OrderID, MenuItemID, Quantity, UnitPrice, Subtotal, SpecialInstructions)

VALUES (p\_OrderID, p\_MenuItemID, p\_Quantity, v\_UnitPrice, v\_Subtotal, p\_SpecialInstructions);

### -- Update the order total

UPDATE `Order`

SET TotalAmount = TotalAmount + v\_Subtotal

WHERE OrderID = p\_OrderID;

END //

CREATE PROCEDURE CompleteOrder(

IN p\_OrderID INT

)

BEGIN

### -- Update the order status

UPDATE `Order`

SET Status = 'Completed'

WHERE OrderID = p\_OrderID;

### -- Create a feedback record

INSERT INTO Feedback (OrderID, CustomerID, FollowUpRequired)

SELECT OrderID, CustomerID, 0

FROM `Order`

WHERE OrderID = p\_OrderID;

### -- Add loyalty points to the customer

UPDATE Customer c

JOIN `Order` o ON c.CustomerID = o.CustomerID

SET c.LoyaltyPoints = c.LoyaltyPoints + FLOOR(o.TotalAmount)

WHERE o.OrderID = p\_OrderID;

END //

DELIMITER ;

## 6.4 Sample Business Analytics Queries

### -- 1. Popular Menu Items by Revenue

SELECT

m.ItemName,

m.Price,

SUM(oi.Quantity) AS TotalOrdered,

SUM(oi.Subtotal) AS TotalRevenue,

c.CategoryName

FROM

OrderItem oi

JOIN MenuItem m ON oi.MenuItemID = m.MenuItemID

JOIN MenuCategory c ON m.CategoryID = c.CategoryID

JOIN `Order` o ON oi.OrderID = o.OrderID

WHERE

o.OrderDate BETWEEN DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY) AND CURRENT\_DATE()

GROUP BY

m.MenuItemID

ORDER BY

TotalRevenue DESC

LIMIT 10;

### -- 2. Customer Loyalty Program Analysis

SELECT

CASE

WHEN LoyaltyPoints = 0 THEN 'New Customer'

WHEN LoyaltyPoints BETWEEN 1 AND 50 THEN 'Bronze Member'

WHEN LoyaltyPoints BETWEEN 51 AND 150 THEN 'Silver Member'

WHEN LoyaltyPoints BETWEEN 151 AND 300 THEN 'Gold Member'

ELSE 'Platinum Member'

END AS LoyaltyTier,

COUNT(\*) AS CustomerCount,

AVG(TotalSpent) AS AvgSpent,

AVG(VisitCount) AS AvgVisits

FROM (

SELECT

c.CustomerID,

c.LoyaltyPoints,

SUM(o.TotalAmount) AS TotalSpent,

COUNT(o.OrderID) AS VisitCount

FROM

Customer c

LEFT JOIN `Order` o ON c.CustomerID = o.CustomerID

GROUP BY

c.CustomerID

) AS CustomerStats

GROUP BY

LoyaltyTier

ORDER BY

AVG(LoyaltyPoints);

# 7. Analysis and Design Artifacts

## 7.1 Individual Project Definition

This individual project for Ma's Tacos Restaurant focuses on designing and implementing a comprehensive data management system to enhance customer engagement, streamline operations, and provide valuable business insights. The project aligns with the overall class project while addressing the specific needs of a family-owned Mexican restaurant business.

The project definition encompasses several key components:

### Customer Engagement Management

* + Development of a loyalty program to track and reward customer visits
  + Implementation of a reservation system with time slot management
  + Creation of a feedback collection mechanism to gather and analyze customer opinions
  + Integration of customer data across all touchpoints for a unified view

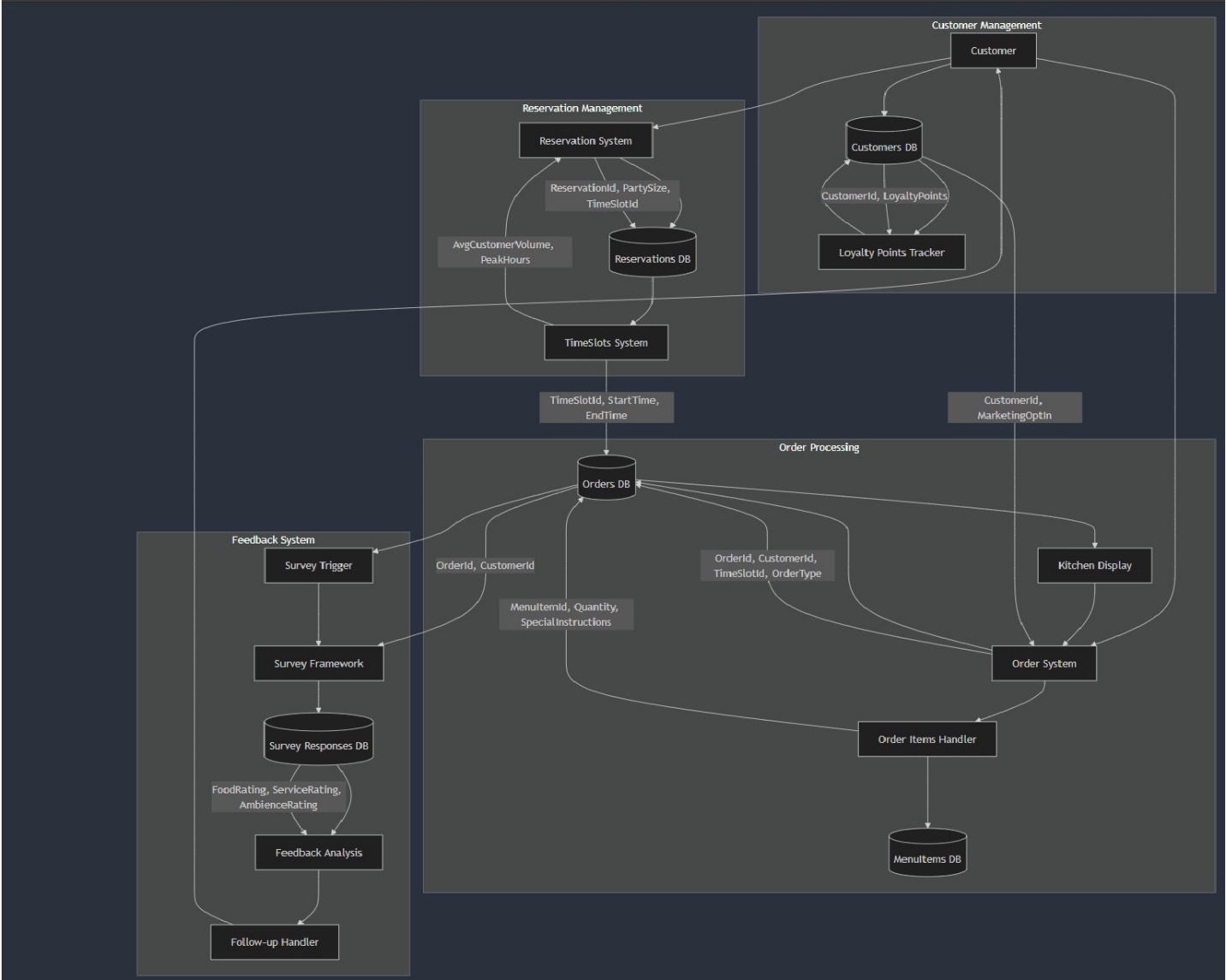
### Order Processing and Analysis

* + Design of an order tracking system with multiple status stages
  + Menu item popularity tracking and analysis
  + Payment type tracking and analysis
  + Order history maintenance for customer relationship management

### Business Intelligence Components

* + Development of analytics dashboards for management decision-making
  + Implementation of reporting tools for operational insights
  + Creation of data-driven marketing capabilities based on customer preferences
  + Inventory management insights based on order patterns

## 7.2 Data Flow Diagram

****

The Data Flow Diagram illustrates how information flows between various system components. As shown in the provided DFD from the original document, the system features several key data flows:

## Key Data Flow Processes:

### Customer Information Flow

* + Customer data enters the system through reservation and order processes
  + Customer profile information is stored in the Customer Database
  + Customer behaviors and preferences feed the loyalty tracking system
  + Customer feedback flows from completed orders to the feedback system

### Order Processing Flow

* + Orders flow from the customer to the order processing system
  + Order details route to the kitchen display system
  + Order items impact inventory levels
  + Completed orders trigger payment processing and feedback collection

### Reservation Management Flow

* + Reservation requests flow to the reservation system
  + Reservation system interfaces with table management and time slot allocation
  + Confirmed reservations may lead to orders
  + Reservation data feeds customer database for history tracking

### Analytics Data Flow

* + Transaction data feeds into analytics systems
  + Customer feedback flows to management reporting
  + Menu item popularity data informs marketing strategies
  + Operational metrics support management decision-making

## 7.3.1 Reservation State Transitions

### States:

* **Pending**: Initial state when reservation is first created
* **Confirmed**: Reservation has been accepted by the restaurant
* **Seated**: Customers have arrived and been seated at their table
* **Completed**: Dining experience has concluded
* **Cancelled**: Reservation was cancelled (from any prior state)

### Transitions:

* **Confirm**: Staff confirms the reservation request
* **Seat**: Customers arrive and are seated at their assigned table
* **Complete**: Customers finish dining and leave
* **Cancel**: Reservation is cancelled (by customer or timeout)

## 7.3.2 Order State Transitions

### States:

* **New**: Order has been placed but not yet processed
* **InProgress**: Kitchen is preparing the order
* **Ready**: Food is prepared and ready to be served
* **Served**: Food has been delivered to the customer
* **Completed**: Order has been paid for
* **Cancelled**: Order was cancelled (from any prior state)

### Transitions:

* **Accept**: Kitchen accepts the order for preparation
* **Ready**: Kitchen marks order as ready for serving
* **Serve**: Server delivers food to customer
* **Complete**: Customer pays for the order
* **Cancel**: Order is cancelled (by customer or staff)

## 7.3.3 Feedback Collection State Transitions

### States:

* **Pending**: Survey has been sent but not yet completed
* **Submitted**: Customer has submitted feedback
* **Analyzed**: Normal feedback has been reviewed
* **Follow-Up**: Low rating feedback requires management attention
* **Resolved**: Follow-up actions have been completed

### Transitions:

* **Submit**: Customer completes the feedback form
* **Low Rating**: System detects rating below threshold
* **Resolve**: Management has addressed the issue

## 7.4 Use Case Diagram

The Use Case Diagram depicts the interactions between system actors and the key functions of the Ma's Tacos Restaurant System.

### Primary Actors:

* Customers
* Staff (servers, hosts, kitchen)
* Restaurant Managers

### Key Use Cases:

#### For Customers:

* + Make Reservation
  + Place Order
  + Provide Feedback
  + View/Redeem Loyalty Points

#### For Staff:

* + Process Reservation
  + Fulfill Order
  + Review Feedback
  + Update Loyalty Program

#### For Managers:

* + Generate Reports
  + Analyze Customer Data
  + Manage Menu Items
  + Configure Time Slots

## Architecture Diagram:

A diagram of a computer

AI-generated content may be incorrect.

# 8. Naming Conventions and Standards

## 8.1 Database Object Naming Conventions

## 8.1.1 Tables

* Tables use singular nouns in PascalCase format (e.g., Customer not Customers)
* Reserved words used as table names are enclosed in backticks (e.g., `Order`, `Table`)
* Consistency in plurality is maintained across all tables (singular form throughout)
* All tables must have a clear, descriptive name that indicates its primary purpose
* Abbreviations are avoided unless they are widely recognized in the restaurant industry context

#### Examples of good table names:

* Customer (not Customers or customer)
* MenuItem (not MenuItems or Menu\_Item)
* Reservation (not Reservations or reservation)

## 8.1.2 Columns

* Column names use PascalCase format (e.g., FirstName, LoyaltyPoints)
* Primary keys follow the pattern [TableName]ID (e.g., CustomerID, OrderID)
* Foreign keys maintain the same name as the primary key they reference
* Boolean fields use an "Is" or similar descriptive prefix (e.g., IsActive, OptInMarketing)
* Date and time fields clearly indicate their purpose (e.g., CreateDate, OrderDate)
* Enumeration fields use descriptive names (e.g., Status, PaymentType)
* Abbreviations are avoided in column names unless widely understood

#### Examples of good column names:

* FirstName (not fname or first\_name)
* TotalAmount (not total or amount)
* IsSpecial (not special or is\_special)

## 8.1.3 Constraints and Indexes

* Primary keys: PK\_[TableName] (e.g., PK\_Customer)
* Foreign keys: FK\_[SourceTable]\_[ReferencedTable] (e.g., FK\_Order\_Customer)
* Unique constraints: UQ\_[TableName]\_[Column(s)] (e.g., UQ\_Customer\_Email)
* Check constraints: CK\_[TableName]\_[Rule] (e.g., CK\_MenuItem\_Price\_Positive)
* Indexes: idx\_[tablename]\_[column(s)] (e.g., idx\_customer\_email)
* Default constraints: DF\_[TableName]\_[Column] (e.g., DF\_Order\_OrderDate)

#### Examples of good index names:

* idx\_customer\_lastname\_firstname (for searching customers by name)
* idx\_order\_orderdate (for filtering orders by date)
* idx\_reservation\_status (for filtering reservations by status)

## 8.2 SQL Coding Standards

### 8.2.1 SQL Keywords and Formatting

* All SQL keywords are written in UPPERCASE (e.g., SELECT, FROM, WHERE)
* Each major clause begins on a new line for readability
* Indentation of 4 spaces is used for nested operations and sub-queries
* Consistent comma placement (after column names) is maintained throughout
* Aliases are used for all tables in joins for clarity and brevity (e.g., c for Customer)
* Column and table names maintain case sensitivity as defined in the database schema
* Parentheses are used to clearly indicate the order of operations in complex conditions
* Comments are included for complex queries to explain their purpose and logic

### 8.2.2 Query Structure Example

SELECT

c.FirstName,

c.LastName,

o.OrderDate,

o.TotalAmount

FROM

Customer c

JOIN `Order` o ON c.CustomerID = o.CustomerID

LEFT JOIN Reservation r ON o.ReservationID = r.ReservationID

WHERE

o.OrderDate > '2023-01-01'

AND c.LoyaltyPoints > 100

AND (

o.Status = 'Completed'

OR o.Status = 'Served'

)

ORDER BY

o.OrderDate DESC,

o.TotalAmount DESC;

### 8.2.3 Stored Procedures

* Names use PascalCase in a verb-noun pattern (e.g., CreateOrder, GetCustomerOrders)
* Parameters are prefixed with "p\_" (e.g., p\_CustomerID, p\_OrderDate)
* Variables are prefixed with "v\_" (e.g., v\_OrderTotal, v\_CurrentStatus)
* Consistent error handling with standardized messages is implemented
* Return values and output parameters are clearly documented in header comments
* Complex logic is broken into well-named sections with explanatory comments
* All stored procedures include a header comment block explaining:
  + Purpose of the procedure
  + Parameters and their usage
  + Return values or result sets
  + Any side effects
  + Author and creation date

#### Example stored procedure header:

/\*\*

\* CreateOrder - Creates a new order for a customer

\*

\* @param p\_CustomerID - The ID of the customer placing the order

\* @param p\_ReservationID - Optional reservation ID (NULL for walk-ins)

\* @param p\_PaymentType - The payment method used

\*

\* @return The newly created order ID

\*

\* @author John Smith

\* @date 2023-09-15

\* @version 1.0

\*/

CREATE PROCEDURE CreateOrder(

IN p\_CustomerID INT,

IN p\_ReservationID INT,

IN p\_PaymentType VARCHAR(20)

)

BEGIN

-- Procedure body

END;

### 8.2.4 Triggers

* Names follow the pattern TR\_[TableName]\_[Event] (e.g., TR\_Order\_AfterUpdate)
* Include comments explaining the trigger's purpose and logic
* Minimize complexity within triggers to maintain performance
* Include error handling to prevent triggering failures from affecting normal operations

### 8.2.5 Views

* Names use PascalCase and begin with "V\_" prefix (e.g., V\_CustomerOrders)
* Include comments explaining the view's purpose and usage
* Select only necessary columns to minimize performance impact
* Avoid complex joins or calculations in views where possible

## 8.3 Data Modeling Standards

### 8.3.1 Entity Relationship Diagram

* Crow's Foot notation is used consistently for relationships
* Entities are represented as rectangles with clear borders
* Attributes are listed within entity rectangles
* Primary keys are indicated with "PK" designation
* Foreign keys are indicated with "FK" designation
* Relationship lines are labeled with descriptive verbs or phrases
* Cardinality is clearly marked (1:1, 1:N, N:M)
* Optional relationships are indicated with dashed lines
* Mandatory relationships are indicated with solid lines
* Entity names match table names in the database schema

### 8.3.2 Data Types

* Consistent data types are used across related fields
* Character fields use VARCHAR with appropriate size limits
* Monetary values use DECIMAL(10,2) for standardized precision
* Date and time fields use appropriate types based on required granularity:
  + DATE for date-only values
  + TIME for time-only values
  + DATETIME for combined date and time values
* Boolean values use TINYINT(1) consistently
* ENUM types are used for fields with predefined values
* INT is used for numerical IDs and counts
* TEXT is used for unlimited-length string data
* Appropriate size limits are applied to all variable-length fields

### 8.3.3 Normalization

* All tables are designed to Third Normal Form (3NF) standards
* Any denormalization for performance is documented with clear rationale
* Primary and foreign key relationships enforce referential integrity
* All tables have a proper primary key
* No multi-valued attributes exist in any table
* No repeating groups exist in any table
* Non-key attributes are dependent on the entire primary key

## 8.4 Documentation Standards

### 8.4.1 Data Dictionary

Each entity and attribute is documented with:

* Technical name exactly matching the database schema
* Business definition in plain language
* Data type and constraints
* Valid values or range of acceptable values
* Default value (if any)
* Business rules governing the data element
* Source of data (system of origin)
* Usage notes and examples

#### Example for Customer entity:

Entity: Customer

Business Definition: Individuals who patronize the restaurant

Primary Key: CustomerID

Attributes:

##### - CustomerID

Definition: Unique identifier for the customer

Data Type: INT, AUTO\_INCREMENT

Constraints: PRIMARY KEY

Usage: Used as a foreign key in Orders, Reservations, and Feedback

##### - FirstName

Definition: Customer's given name

Data Type: VARCHAR(50)

Constraints: NOT NULL

Usage: Used for addressing customers and in search functions

##### - Email

Definition: Customer's email address

Data Type: VARCHAR(100)

Constraints: UNIQUE

Business Rules: Must be valid email format (contains @ and domain)

Usage: Used for sending confirmations and marketing communications

##### - LoyaltyPoints

Definition: Points earned through purchases

Data Type: INT

Default Value: 0

Business Rules: Increased by 1 point per dollar spent (rounded down)

Usage: Used to determine loyalty tier and available rewards

### 8.4.2 Model Versioning

* All models include version number in format X.Y (e.g., 1.0, 1.1, 2.0)
* Major version increments (X) indicate significant structural changes
* Minor version increments (Y) indicate smaller modifications or additions
* Change history is documented with:
  + Date of change
  + Nature of change
  + Reason for change
  + Person responsible for change
* Date of last update is included in all model documentation
* Author of changes is recorded
* Major versions align with project milestones
* Model changes are subject to peer review before implementation
* Version history is maintained in a centralized document

### 8.4.3 Security Classification

Data elements are classified by sensitivity level:

* **Public**: No sensitivity (e.g., menu items, business hours)
  + Can be freely shared and displayed
  + No special protection required
* **Internal**: Business operational data (e.g., reservations, table assignments)
  + Accessible to all employees but not customers
  + Protected from external access
* **Confidential**: Customer personal data (e.g., email, phone, address)
  + Limited access to authorized personnel
  + Requires secure transmission and storage
  + Subject to privacy regulations and policies
* **Restricted**: Payment and financial information
  + Highest level of protection required
  + Access limited to specific roles
  + Subject to PCI compliance requirements
  + Encrypted in transit and at rest

## 8.5 Implementation Standards

### 8.5.1 Database Creation

* Character set: utf8mb4 (to support all Unicode characters including emoji)
* Collation: utf8mb4\_unicode\_ci (case-insensitive Unicode collation)
* Storage engine: InnoDB for all tables (for transaction support and referential integrity)
* Auto-increment starting value: 1001 (to distinguish from test data)
* Properly commented DDL scripts with version information
* Script execution order clearly documented
* Database creation scripts include:
  + Database creation
  + User creation with appropriate permissions
  + Table creation with constraints
  + Index creation
  + Stored procedure and view creation
  + Initial data population

### 8.5.2 Performance Considerations

* Indexes on all frequently queried columns
* Compound indexes for multi-column filtering conditions
* Covering indexes for frequently run queries
* VARCHAR fields sized appropriately for expected data
* TEXT type used only when necessary for unlimited length strings
* Normalization to 3NF for data integrity
* Strategic denormalization documented where performance requires it
* EXPLAIN plans reviewed for all complex queries
* Query execution plan analysis for stored procedures
* Appropriate use of connection pooling
* Efficient transaction management to minimize lock contention

These standards ensure consistency across all database objects, facilitate collaboration among team members, and promote maintainability of the system over time. They provide a clear framework for development and documentation that aligns with industry best practices and the specific needs of the Ma's Tacos restaurant data system.

# 9. Conclusion

The Ma's Tacos Restaurant Data System provides a comprehensive solution for managing customer engagement, orders, reservations, and analytics for the restaurant. The system is designed with scalability in mind, allowing for future enhancements and potential adaptation for other restaurant businesses.

Key features of the solution include:

* Reservation and table management system
* Order processing with kitchen integration
* Customer loyalty program
* Feedback collection and analysis
* Menu performance analytics
* Payment processing and reconciliation

The database design follows normalization principles to ensure data integrity while optimizing for the most common query patterns. The physical implementation includes appropriate indexes and constraints to maintain performance as the data volume grows.

By implementing this system, Ma's Tacos will gain valuable insights into customer preferences, operational efficiency, and marketing effectiveness, ultimately leading to increased customer satisfaction and business growth.