P4. Database Schema Implementation

```sql
--Insert DB

USE RestaurantDB;
GO

-- Insert data in the correct order to avoid FK violations

-- First insert restaurant without ManagerID
INSERT INTO RESTAURANT (Name, Location, ContactNumber, ManagerID, Capacity)
VALUES ('The Gourmet Spot', '123 Main Street', '555-0101', NULL, 150);

-- Now insert manager
INSERT INTO STAFF (RestaurantID, FullName, ContactNumber, Role)
VALUES (1, 'Alice Johnson', '555-0001', 'Manager');

-- Update restaurant to set the ManagerID
UPDATE RESTAURANT SET ManagerID = 1 WHERE RestaurantID = 1;

-- Insert remaining STAFF
INSERT INTO STAFF (RestaurantID, FullName, ContactNumber, Role) VALUES
(1, 'Bob Smith', '555-0002', 'Chef'),
(1, 'Carol White', '555-0003', 'Waiter'),
(1, 'Dave Brown', '555-0004', 'Bartender'),
(1, 'Eve Davis', '555-0005', 'Host'),
(1, 'Frank Wilson', '555-0006', 'Cashier'),
(1, 'Grace Lee', '555-0007', 'Sous Chef'),

(1, 'Henry Moore', '555-0008', 'Waiter'),

(1, 'Ivy Taylor', '555-0009', 'Chef'),

(1, 'Jack Martinez', '555-0010', 'Host');


-- CUSTOMER (10 rows)

INSERT INTO CUSTOMER (FullName, Email, Contact, Address) VALUES

('John Doe', 'john@email.com', '555-0201', '123 Elm St'),

('Jane Smith', 'jane@email.com', '555-0202', '456 Oak St'),

('Mike Brown', 'mike@email.com', '555-0203', '789 Pine St'),

('Sarah Wilson', 'sarah@email.com', '555-0204', '321 Maple St'),

('Tom Davis', 'tom@email.com', '555-0205', '654 Cedar St'),

('Lucy Miller', 'lucy@email.com', '555-0206', '987 Birch St'),

('Bill Taylor', 'bill@email.com', '555-0207', '135 Spruce St'),

('Anna Clark', 'anna@email.com', '555-0208', '246 Ash St'),

('Chris Lee', 'chris@email.com', '555-0209', '369 Fir St'),

('Patty Lewis', 'patty@email.com', '555-0210', '159 Redwood St');


-- MENU (10 rows)

INSERT INTO MENU (RestaurantID, Name, Description) VALUES

(1, 'Breakfast', 'Morning specialties'),

(1, 'Lunch', 'Midday meals'),

(1, 'Dinner', 'Evening gourmet'),

(1, 'Drinks', 'Beverages'),

(1, 'Desserts', 'Sweet treats'),

(1, 'Kids', 'Children''s meals'),
(1, 'Vegetarian', 'Plant-based options'),

(1, 'Specials', 'Chef''s specials'),

(1, 'Happy Hour', 'Discount drinks'),

(1, 'Seasonal', 'Limited-time offers');


-- DISH (10 rows)

INSERT INTO DISH (MenuID, Name, Price, Ingredients, PreparationTime, IsVegetarian)
VALUES

(1, 'Pancakes', 8.99, 'Flour, eggs, milk', 15, 1),

(2, 'Caesar Salad', 12.50, 'Lettuce, croutons', 10, 0),

(3, 'Steak', 29.99, 'Beef, seasoning', 25, 0),

(4, 'Mojito', 9.99, 'Rum, mint', 5, 1),

(5, 'Cheesecake', 7.99, 'Cream cheese', 20, 1),

(6, 'Chicken Tenders', 6.99, 'Chicken, breading', 12, 0),

(7, 'Veggie Burger', 10.99, 'Beans, veggies', 15, 1),

(8, 'Soup', 6.50, 'Vegetables, broth', 15, 1),

(9, 'Wine', 12.99, 'Grapes', 2, 1),

(10, 'Salmon', 24.99, 'Salmon, herbs', 20, 0);


-- SUPPLIER (10 rows)

INSERT INTO SUPPLIER (Name, ContactPerson, ContactNumber, SuppliesCategory,
Address) VALUES

('Fresh Farms', 'John Supplier', '555-0301', 'Produce', '100 Farm Rd'),

('Meat Masters', 'Bob Butcher', '555-0302', 'Meat', '200 Ranch Rd'),

('Dairy Delight', 'Clara Cow', '555-0303', 'Dairy', '300 Milk St'),

('Beverage Co', 'Joe Juicer', '555-0304', 'Beverages', '400 Soda Ave'),

('Bakery Goods', 'Bread Baker', '555-0305', 'Baked Goods', '500 Flour Ln'),
('Seafood Ltd', 'Finn Fisher', '555-0306', 'Seafood', '600 Ocean Dr'),
('Spice World', 'Sally Spice', '555-0307', 'Spices', '700 Seasoning Blvd'),
('Frozen Foods Inc', 'Icy Cold', '555-0308', 'Frozen Goods', '800 Freeze Cir'),
('Organic Oasis', 'Olive Organic', '555-0309', 'Organic Produce', '900 Green Way'),
('Utensil Supply', 'Peter Pan', '555-0310', 'Kitchenware', '1000 Tool St');

-- INVENTORY (10 rows)
INSERT INTO INVENTORY (RestaurantID, SupplierID, IngredientName, StockLevel,
LastRestockedDate, ReorderThreshold) VALUES
(1, 1, 'Tomatoes', 50, '2023-10-01', 20),
(1, 2, 'Beef', 30, '2023-10-02', 15),
(1, 3, 'Milk', 100, '2023-10-03', 50),
(1, 4, 'Soda', 200, '2023-10-04', 100),
(1, 5, 'Flour', 150, '2023-10-05', 75),
(1, 6, 'Salmon', 40, '2023-10-06', 20),
(1, 7, 'Paprika', 30, '2023-10-07', 10),
(1, 8, 'Frozen Fries', 80, '2023-10-08', 40),
(1, 9, 'Organic Kale', 60, '2023-10-09', 30),
(1, 10, 'Forks', 200, '2023-10-10', 100);

-- RESERVATION (10 rows)
INSERT INTO RESERVATION (CustomerID, RestaurantID, ReservationDate, TimeSlot,
PartySize, Status) VALUES
(1, 1, '2023-10-01', '18:00', 4, 'Confirmed'),
(2, 1, '2023-10-02', '19:30', 2, 'Confirmed'),
(3, 1, '2023-10-03', '20:00', 6, 'Pending'),
(4, 1, '2023-10-04', '17:45', 5, 'Confirmed'),
(5, 1, '2023-10-05', '19:00', 3, 'Canceled'),

(6, 1, '2023-10-06', '18:30', 4, 'Confirmed'),

(7, 1, '2023-10-07', '20:15', 2, 'Pending'),

(8, 1, '2023-10-08', '19:00', 8, 'Confirmed'),

(9, 1, '2023-10-09', '18:45', 6, 'Confirmed'),

(10, 1, '2023-10-10', '21:00', 4, 'Pending');


-- ORDERS (10 rows)

INSERT INTO ORDERS (CustomerID, RestaurantID, ReservationID, OrderDate) VALUES

(1, 1, 1, GETDATE()),

(2, 1, NULL, GETDATE()),

(3, 1, 3, GETDATE()),

(4, 1, 4, GETDATE()),

(5, 1, NULL, GETDATE()),

(6, 1, 6, GETDATE()),

(7, 1, NULL, GETDATE()),

(8, 1, 8, GETDATE()),

(9, 1, 9, GETDATE()),

(10, 1, NULL, GETDATE());


-- ORDER_DETAILS (10 rows)

INSERT INTO ORDER_DETAILS (OrderID, DishID, Quantity, Subtotal) VALUES

(1, 1, 2, (SELECT Price FROM DISH WHERE DishID = 1) * 2),

(2, 2, 1, (SELECT Price FROM DISH WHERE DishID = 2)),

(3, 3, 1, (SELECT Price FROM DISH WHERE DishID = 3)),

(4, 4, 3, (SELECT Price FROM DISH WHERE DishID = 4) * 3),
```

```sql
(5, 5, 2, (SELECT Price FROM DISH WHERE DishID = 5) * 2),
(6, 6, 4, (SELECT Price FROM DISH WHERE DishID = 6) * 4),
(7, 7, 1, (SELECT Price FROM DISH WHERE DishID = 7)),
(8, 8, 2, (SELECT Price FROM DISH WHERE DishID = 8) * 2),
(9, 9, 5, (SELECT Price FROM DISH WHERE DishID = 9) * 5),
(10, 10, 1, (SELECT Price FROM DISH WHERE DishID = 10));

-- PAYMENT (10 rows)
INSERT INTO PAYMENT (OrderID, PaymentType, Amount) VALUES
(1, 'Credit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 1)),
(2, 'Cash', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 2)),
(3, 'Debit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 3)),
(4, 'Credit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 4)),
(5, 'Cash', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 5)),
(6, 'Debit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 6)),
(7, 'Credit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 7)),
(8, 'Cash', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 8)),
(9, 'Debit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 9)),
(10, 'Credit Card', (SELECT SUM(Subtotal) FROM ORDER_DETAILS WHERE OrderID = 10));

-- CARD_PAYMENT - Only for credit/debit card payments
INSERT INTO CARD_PAYMENT (PaymentID, CardNumber, ExpirationDate, CVV) VALUES
(1, '4111111111111111', '2025-12-01', '123'),
(3, '5555555555554444', '2024-10-01', '456'),
(4, '378282246310005', '2026-05-01', '789'),
(6, '6011111111111117', '2025-08-01', '321'),
(7, '3530111333300000', '2024-12-01', '654'),

(9, '5105105105105100', '2025-07-01', '987'),

(10, '4111111111111111', '2025-11-01', '111');


-- FEEDBACK (10 rows)

INSERT INTO FEEDBACK (CustomerID, OrderID, FeedbackType, Rating, Comments)
VALUES

(1, 1, 'Positive', 5, 'Excellent service'),

(2, 2, 'Neutral', 3, 'Average experience'),

(3, 3, 'Negative', 2, 'Slow delivery'),

(4, 4, 'Positive', 4, 'Good food'),

(5, 5, 'Negative', 1, 'Cold meal'),

(6, 6, 'Positive', 5, 'Will return'),

(7, 7, 'Neutral', 3, 'Decent'),

(8, 8, 'Positive', 4, 'Friendly staff'),

(9, 9, 'Positive', 5, 'Amazing dishes'),

(10, 10, 'Negative', 2, 'Overpriced');


-- SHIFTS (10 rows) - Fixed shift data

INSERT INTO SHIFTS (StaffID, ShiftDate, StartTime, EndTime) VALUES

(1, '2023-10-01', '08:00', '16:00'),

(2, '2023-10-01', '09:00', '17:00'),

(3, '2023-10-02', '10:00', '18:00'),

(4, '2023-10-02', '12:00', '20:00'),

(5, '2023-10-03', '07:00', '15:00'),

(6, '2023-10-03', '14:00', '22:00'),
(7, '2023-10-04', '08:30', '16:30'),

(8, '2023-10-04', '11:00', '19:00'),

(9, '2023-10-05', '13:00', '21:00'),

(1, '2023-10-06', '09:30', '17:30');


INSERT INTO RESTAURANT (Name, Location, ContactNumber, ManagerID, Capacity)
VALUES

('Food Haven', '123 Main St', '123-456-7890', NULL, 50),

('Gourmet House', '456 Park Ave', '234-567-8901', NULL, 75),

('Tasty Treats', '789 Oak St', '345-678-9012', NULL, 100),

('The Dine-In', '321 Maple St', '456-789-0123', NULL, 60),

('Spice World', '654 Pine St', '567-890-1234', NULL, 80),

('Urban Eats', '987 Birch St', '678-901-2345', NULL, 90),

('Sizzling Grill', '111 Cedar St', '789-012-3456', NULL, 120),

('Bistro Bliss', '222 Cherry St', '890-123-4567', NULL, 40),

('Culinary Haven', '333 Walnut St', '901-234-5678', NULL, 55),

('Fusion Flavors', '444 Chestnut St', '012-345-6789', NULL, 70);


GO

--CREATE DB

-- Drop the database if it exists

IF EXISTS (SELECT name FROM sys.databases WHERE name = 'RestaurantDB')

   DROP DATABASE RestaurantDB;

GO

CREATE DATABASE RestaurantDB;
```

```
GO

USE RestaurantDB;

GO


-- Drop tables if they exist (in correct order to avoid FK issues)

IF OBJECT_ID('dbo.CARD_PAYMENT', 'U') IS NOT NULL DROP TABLE dbo.CARD_PAYMENT;

IF OBJECT_ID('dbo.PAYMENT', 'U') IS NOT NULL DROP TABLE dbo.PAYMENT;

IF OBJECT_ID('dbo.FEEDBACK', 'U') IS NOT NULL DROP TABLE dbo.FEEDBACK;

IF OBJECT_ID('dbo.ORDER_DETAILS', 'U') IS NOT NULL DROP TABLE dbo.ORDER_DETAILS;

IF OBJECT_ID('dbo.ORDERS', 'U') IS NOT NULL DROP TABLE dbo.ORDERS;

IF OBJECT_ID('dbo.RESERVATION', 'U') IS NOT NULL DROP TABLE dbo.RESERVATION;

IF OBJECT_ID('dbo.SHIFTS', 'U') IS NOT NULL DROP TABLE dbo.SHIFTS;

IF OBJECT_ID('dbo.INVENTORY', 'U') IS NOT NULL DROP TABLE dbo.INVENTORY;

IF OBJECT_ID('dbo.DISH', 'U') IS NOT NULL DROP TABLE dbo.DISH;

IF OBJECT_ID('dbo.MENU', 'U') IS NOT NULL DROP TABLE dbo.MENU;

IF OBJECT_ID('dbo.SUPPLIER', 'U') IS NOT NULL DROP TABLE dbo.SUPPLIER;

IF OBJECT_ID('dbo.STAFF', 'U') IS NOT NULL DROP TABLE dbo.STAFF;

IF OBJECT_ID('dbo.RESTAURANT', 'U') IS NOT NULL DROP TABLE dbo.RESTAURANT;

IF OBJECT_ID('dbo.CUSTOMER', 'U') IS NOT NULL DROP TABLE dbo.CUSTOMER;

GO


-- First create tables without circular dependency

CREATE TABLE RESTAURANT (

  RestaurantID INT IDENTITY(1,1) PRIMARY KEY,

  Name NVARCHAR(100) NOT NULL,

  Location NVARCHAR(255) NOT NULL,
  ContactNumber NVARCHAR(20) UNIQUE NOT NULL,
  ManagerID INT NULL, -- Changed to NULL to allow initial insert
  Capacity INT NOT NULL CHECK (Capacity > 0)
);

CREATE TABLE STAFF (
  StaffID INT IDENTITY(1,1) PRIMARY KEY,
  RestaurantID INT NOT NULL,
  FullName NVARCHAR(100) NOT NULL,
  ContactNumber NVARCHAR(20) UNIQUE NOT NULL,
  Role NVARCHAR(50) NOT NULL CHECK (Role IN ('Chef', 'Waiter', 'Manager', 'Bartender',
'Host', 'Cashier', 'Sous Chef')),
  FOREIGN KEY (RestaurantID) REFERENCES RESTAURANT(RestaurantID) ON DELETE
CASCADE
);

-- Later we'll add the FK for ManagerID
ALTER TABLE RESTAURANT
ADD CONSTRAINT FK_Restaurant_Manager FOREIGN KEY (ManagerID) REFERENCES
STAFF(StaffID);

-- MENU Table
CREATE TABLE MENU (
  MenuID INT IDENTITY(1,1) PRIMARY KEY,
  RestaurantID INT NOT NULL,
  Name NVARCHAR(100) NOT NULL,
  Description NVARCHAR(MAX),
  ActiveStatus BIT DEFAULT 1,
  FOREIGN KEY (RestaurantID) REFERENCES RESTAURANT(RestaurantID) ON DELETE
CASCADE
);

-- DISH Table
CREATE TABLE DISH (
  DishID INT IDENTITY(1,1) PRIMARY KEY,
  MenuID INT NOT NULL,
  Name NVARCHAR(100) NOT NULL,
  Price DECIMAL(10,2) NOT NULL CHECK (Price > 0),
  Ingredients NVARCHAR(MAX) NOT NULL,
  PreparationTime INT NOT NULL CHECK (PreparationTime > 0),
  IsVegetarian BIT DEFAULT 0,
  IsAvailable BIT DEFAULT 1,
  FOREIGN KEY (MenuID) REFERENCES MENU(MenuID) ON DELETE CASCADE
);

-- CUSTOMER Table
CREATE TABLE CUSTOMER (
  CustomerID INT IDENTITY(1,1) PRIMARY KEY,
  FullName NVARCHAR(100) NOT NULL,
  Email NVARCHAR(100) UNIQUE NOT NULL,
  Contact NVARCHAR(20) UNIQUE NOT NULL,
  Address NVARCHAR(MAX) NOT NULL
);
```

```
-- RESERVATION Table

CREATE TABLE RESERVATION (

   ReservationID INT IDENTITY(1,1) PRIMARY KEY,

   CustomerID INT NOT NULL,

   RestaurantID INT NOT NULL,

   ReservationDate DATE NOT NULL,

   TimeSlot TIME NOT NULL,

   PartySize INT NOT NULL CHECK (PartySize > 0),

   Status NVARCHAR(20) DEFAULT 'Pending' CHECK (Status IN ('Pending', 'Confirmed',
'Canceled')),

   FOREIGN KEY (CustomerID) REFERENCES CUSTOMER(CustomerID) ON DELETE
CASCADE,

   FOREIGN KEY (RestaurantID) REFERENCES RESTAURANT(RestaurantID) ON DELETE
CASCADE

);


-- ORDER Table

CREATE TABLE ORDERS (

   OrderID INT IDENTITY(1,1) PRIMARY KEY,

   CustomerID INT NOT NULL,

   RestaurantID INT NOT NULL,

   ReservationID INT NULL,

   OrderDate DATETIME DEFAULT GETDATE(),

   FOREIGN KEY (CustomerID) REFERENCES CUSTOMER(CustomerID) ON DELETE
CASCADE,
   FOREIGN KEY (RestaurantID) REFERENCES RESTAURANT(RestaurantID) ON DELETE
CASCADE,
   FOREIGN KEY (ReservationID) REFERENCES RESERVATION(ReservationID) ON DELETE
NO ACTION
);

-- ORDER DETAILS Table
CREATE TABLE ORDER_DETAILS (
   OrderDetailID INT IDENTITY(1,1) PRIMARY KEY,
   OrderID INT NOT NULL,
   DishID INT NOT NULL,
   Quantity INT NOT NULL CHECK (Quantity > 0),
   Subtotal DECIMAL(10,2) NOT NULL CHECK (Subtotal >= 0),
   FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID) ON DELETE CASCADE,
   FOREIGN KEY (DishID) REFERENCES DISH(DishID) ON DELETE NO ACTION
);

-- PAYMENT Table
CREATE TABLE PAYMENT (
   PaymentID INT IDENTITY(1,1) PRIMARY KEY,
   OrderID INT NOT NULL,
   PaymentType NVARCHAR(20) NOT NULL CHECK (PaymentType IN ('Credit Card', 'Debit
Card', 'Cash')),
   Amount DECIMAL(10,2) NOT NULL CHECK (Amount > 0),
   TransactionDate DATETIME DEFAULT GETDATE(),
   FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID) ON DELETE CASCADE
);
```

```
-- CARD_PAYMENT Table

CREATE TABLE CARD_PAYMENT (

   PaymentID INT PRIMARY KEY,

   CardNumber NVARCHAR(16) NOT NULL,

   ExpirationDate DATE NOT NULL,

   CVV CHAR(4) NOT NULL CHECK (LEN(CVV) BETWEEN 3 AND 4),

   FOREIGN KEY (PaymentID) REFERENCES PAYMENT(PaymentID) ON DELETE CASCADE

);


-- FEEDBACK Table

CREATE TABLE FEEDBACK (

   FeedbackID INT IDENTITY(1,1) PRIMARY KEY,

   CustomerID INT NOT NULL,

   OrderID INT NOT NULL,

   FeedbackType NVARCHAR(20) NOT NULL CHECK (FeedbackType IN ('Positive', 'Negative',
'Neutral')),

   Rating INT NOT NULL CHECK (Rating BETWEEN 1 AND 5),

   Comments NVARCHAR(MAX),

   FeedbackDate DATE DEFAULT GETDATE(),

   FOREIGN KEY (CustomerID) REFERENCES CUSTOMER(CustomerID) ON DELETE
CASCADE,

   FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID) ON DELETE NO ACTION

);


-- SUPPLIER Table

CREATE TABLE SUPPLIER (
   SupplierID INT IDENTITY(1,1) PRIMARY KEY,

   Name NVARCHAR(100) NOT NULL,

   ContactPerson NVARCHAR(100),

   ContactNumber NVARCHAR(20) UNIQUE NOT NULL,

   SuppliesCategory NVARCHAR(100),

   Address NVARCHAR(MAX) NOT NULL

);


-- INVENTORY Table

CREATE TABLE INVENTORY (

   InventoryID INT IDENTITY(1,1) PRIMARY KEY,

   RestaurantID INT NOT NULL,

   SupplierID INT NOT NULL,

   IngredientName NVARCHAR(100) NOT NULL,

   StockLevel INT NOT NULL CHECK (StockLevel >= 0),

   LastRestockedDate DATE,

   ReorderThreshold INT NOT NULL CHECK (ReorderThreshold > 0),

   FOREIGN KEY (RestaurantID) REFERENCES RESTAURANT(RestaurantID) ON DELETE
CASCADE,

   FOREIGN KEY (SupplierID) REFERENCES SUPPLIER(SupplierID) ON DELETE CASCADE

);


-- SHIFTS Table

CREATE TABLE SHIFTS (

   ShiftID INT IDENTITY(1,1) PRIMARY KEY,

   StaffID INT NOT NULL,
   ShiftDate DATE NOT NULL,

   StartTime TIME NOT NULL,

   EndTime TIME NOT NULL,

   FOREIGN KEY (StaffID) REFERENCES STAFF(StaffID) ON DELETE CASCADE

);



GO
```