Tab 1

INFORMATION MANAGEMENT IN RESTAURANT

Scenario:

A waiter takes an order at a table, and then enters it online via one of the six terminals located in the restaurant dining room. The order is routed to a printer in the appropriate preparation area: the cold item printer if it is a *salad*, the hot-item printer if it is a *hot sandwich* or the bar printer if it is a *drink*. A customer's meal check-listing (bill) the items ordered and the respective prices are automatically generated. This ordering system eliminates the old three-carbon-copy guest check system as well as any problems caused by a waiter's handwriting.

When the kitchen runs out of a food item, the cooks send out an "out of stock" message, which will be displayed on the dining room terminals when waiters try to order that item. This gives the waiters faster feedback, enabling them to give better service to the customers. Other system features aid management in the planning and control of their restaurant business.

The system provides up-to-the-minute information on the food items ordered and breaks out percentages showing sales of each item versus total sales. This helps management plan menus according to customers' tastes. The system also compares the weekly sales totals versus food costs, allowing planning for tighter cost controls. In addition, whenever an order is voided, the reasons for the void are keyed in. This may help later in management decisions, especially if the voids are consistently related to food or service.

Acceptance of the system by the users is exceptionally high since the waiters and waitresses were involved in the selection and design process. All potential users were asked to give their impressions and ideas about the various systems available before one was chosen.

INSTRUCTIONS FOR STUDENTS

1. Review the Scenario Carefully

Read the restaurant ordering system scenario provided in the activity. Identify:

- The **main entities** (tables)
- **Attributes** needed for each table
- **Constraints** such as primary keys, foreign keys, and availability rules

2. Design the Database Structure

Before writing SQL code, you must:

1. Identify **all required tables**
2. Specify **attributes** and **data types**

3. Assign **Primary Keys (PK)** and **Foreign Keys (FK)**

3. Write the SQL CREATE TABLE Scripts

Using MySQL syntax, create the tables in the correct order following these rules:

- Define appropriate **data types** (e.g., INT, VARCHAR, DECIMAL, DATE, etc.)
- Apply **PK, FK, NOT NULL, DEFAULT**, and other necessary constraints
- Include meaningful names for columns and tables
- The SQL script must run **without errors** when executed

---

4. Insert Sample Data Into Each Table

Create **at least 2–3 rows** per table, ensuring:

- All data is **realistic**, relevant, and consistent with the scenario

Place all INSERT statements **after** the CREATE TABLE statements.

vas

PART 1: Understand the Scenario (Very Important)

Before typing SQL, you must understand **what data needs to be stored**.

From the scenario, the system needs to store:

Main things (Entities / Tables)

1. **Waiters** – who takes the order
2. **Restaurant Tables** – where customers sit
3. **Menu Items** – food and drinks
4. **Orders** – one order per customer visit
5. **Order Items** – what items are inside an order
6. **Printers** – where orders are sent (cold, hot, bar)
7. **Voided Orders** – reason why an order/item was voided

PART 2: Identify Tables and Attributes

1. `waiters`

Column	Meaning
waiter_id	Unique ID
waiter_name	Name

2. `restaurant_tables`

Column	Meaning
table_id	Table number
capacity	Number of seats

3. `printers`

Column	Meaning
printer_id	Unique ID
printer_name	Cold, Hot, Bar

4. `menu_items`

Column	Meaning
item_id	Unique item ID
item_name	Name of food/drink
price	Item price
printer_id	Which printer it goes to
in_stock	Available or not

5. `orders`

Column	Meaning
order_id	Unique order
waiter_id	Who took the order
table_id	Which table
order_date	Date/time

6. `order_items`

Column	Meaning
order_item_id	Unique row
order_id	Which order

item_id	Which menu item
quantity	How many

7. `voids`

Column	Meaning
void_id	Unique ID
order_item_id	Voided item
reason	Why it was voided

PART 3: SQL — CREATE TABLE Statements

⚠️ **Type these in the exact order**

STEP 1: Create Database

```
CREATE DATABASE restaurant_system;
USE restaurant_system;
```

STEP 2: Create `waiters`

```
CREATE TABLE waiters (
    waiter_id INT AUTO_INCREMENT PRIMARY KEY,
    waiter_name VARCHAR(100) NOT NULL
);
```

STEP 3: Create `restaurant_tables`

```
CREATE TABLE restaurant_tables (
    table_id INT AUTO_INCREMENT PRIMARY KEY,
    capacity INT NOT NULL
);
```

STEP 4: Create `printers`

```
CREATE TABLE printers (
    printer_id INT AUTO_INCREMENT PRIMARY KEY,
    printer_name VARCHAR(50) NOT NULL
);
```

STEP 5: Create `menu_items`

```
CREATE TABLE menu_items (
    item_id INT AUTO_INCREMENT PRIMARY KEY,
    item_name VARCHAR(100) NOT NULL,
    price DECIMAL(6,2) NOT NULL,
    printer_id INT,
    in_stock BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (printer_id) REFERENCES printers(printer_id)
);
```

STEP 6: Create `orders`

```
CREATE TABLE orders (
    order_id INT AUTO_INCREMENT PRIMARY KEY,
    waiter_id INT,
    table_id INT,
    order_date DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (waiter_id) REFERENCES waiters(waiter_id),
    FOREIGN KEY (table_id) REFERENCES restaurant_tables(table_id)
);
```

STEP 7: Create `order_items`

```
CREATE TABLE order_items (
    order_item_id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT,
    item_id INT,
    quantity INT NOT NULL,
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (item_id) REFERENCES menu_items(item_id)
);
```

STEP 8: Create `voids`

```
CREATE TABLE voids (
    void_id INT AUTO_INCREMENT PRIMARY KEY,
    order_item_id INT,
    reason VARCHAR(255),
    FOREIGN KEY (order_item_id) REFERENCES order_items(order_item_id)
);
```

✅ **If all tables are created without errors, you did it correctly**

PART 4: Insert Sample Data

⚠️ **Always insert parent tables first**

---

1. Insert Waiters

```
INSERT INTO waiters (waiter_name) VALUES
('Juan Dela Cruz'),
('Maria Santos');
```

---

2. Insert Restaurant Tables

```
INSERT INTO restaurant_tables (capacity) VALUES
(4),
(6);
```

---

3. Insert Printers

```
INSERT INTO printers (printer_name) VALUES
('Cold Item Printer'),
('Hot Item Printer'),
('Bar Printer');
```

---

4. Insert Menu Items

```
INSERT INTO menu_items (item_name, price, printer_id, in_stock) VALUES
('Caesar Salad', 150.00, 1, TRUE),
('Hot Sandwich', 200.00, 2, TRUE),
('Iced Tea', 80.00, 3, TRUE);
```

---

5. Insert Orders

```
INSERT INTO orders (waiter_id, table_id) VALUES
(1, 1),
(2, 2);
```

---

6. Insert Order Items

```
INSERT INTO order_items (order_id, item_id, quantity) VALUES
(1, 1, 1),
(1, 3, 2),
(2, 2, 1);
```

---

7. Insert Voided Item

```
INSERT INTO voids (order_item_id, reason) VALUES
```

```
(3, 'Customer changed mind');
```

Tab 3

PART 0: What You Need

- XAMPP installed on your computer
- MySQL included in XAMPP
- A web browser (Chrome is fine)

PART 1: Open XAMPP and Start MySQL

STEP 1: Open XAMPP Control Panel

1. Click **Start Menu**
2. Type **XAMPP**
3. Click **XAMPP Control Panel**

You should now see:

- Apache
- MySQL
- FileZilla
- Mercury
- Tomcat

STEP 2: Start MySQL & APACHE

1. Look for **MySQL**
2. Click the **Start** button next to it
3. Wait until the background turns **green**

✅ MySQL is now running

PART 2: Open phpMyAdmin (Database Interface)

STEP 3: Open phpMyAdmin

1. In XAMPP Control Panel, click **Admin** beside **MySQL**
2. Your browser opens automatically
3. You should see **phpMyAdmin dashboard**

If not:

- Open browser
- Type in address bar:

```
http://localhost/phpmyadmin
```

PART 3: Create the Database

STEP 4: Create a New Database

1. On the **left side**, click **New**
2. In **Database name**, type:

```
restaurant_system
```

3. Click **Create**

✅ Database is created

PART 4: Open the SQL Tab

STEP 5: Go to SQL Editor

1. Click **restaurant_system** on the left panel
2. Click the **SQL** tab at the top

This is where you will type SQL code.

PART 5: Create Tables (TYPE EXACTLY THIS)

⚠️ **Copy and paste ONE BLOCK AT A TIME**
⚠️ **Click "Go" after each block**

STEP 6: Create `waiters`

```
CREATE TABLE waiters (
   waiter_id INT AUTO_INCREMENT PRIMARY KEY,
   waiter_name VARCHAR(100) NOT NULL
);
```

➡️ Click **Go**

STEP 7: Create `restaurant_tables`

```
CREATE TABLE restaurant_tables (
   table_id INT AUTO_INCREMENT PRIMARY KEY,
   capacity INT NOT NULL
);
```

➡️ Click **Go**

STEP 8: Create `printers`

```
CREATE TABLE printers (
   printer_id INT AUTO_INCREMENT PRIMARY KEY,
   printer_name VARCHAR(50) NOT NULL
);
```

➡ Click **Go**

STEP 9: Create `menu_items`

```
CREATE TABLE menu_items (
   item_id INT AUTO_INCREMENT PRIMARY KEY,
   item_name VARCHAR(100) NOT NULL,
   price DECIMAL(6,2) NOT NULL,
   printer_id INT,
   in_stock BOOLEAN DEFAULT TRUE,
   FOREIGN KEY (printer_id) REFERENCES printers(printer_id)
);
```

➡ Click **Go**

STEP 10: Create `orders`

```
CREATE TABLE orders (
   order_id INT AUTO_INCREMENT PRIMARY KEY,
   waiter_id INT,
   table_id INT,
   order_date DATETIME DEFAULT CURRENT_TIMESTAMP,
   FOREIGN KEY (waiter_id) REFERENCES waiters(waiter_id),
   FOREIGN KEY (table_id) REFERENCES restaurant_tables(table_id)
);
```

➡ Click **Go**

STEP 11: Create `order_items`

```
CREATE TABLE order_items (
   order_item_id INT AUTO_INCREMENT PRIMARY KEY,
   order_id INT,
   item_id INT,
   quantity INT NOT NULL,
   FOREIGN KEY (order_id) REFERENCES orders(order_id),
   FOREIGN KEY (item_id) REFERENCES menu_items(item_id)
);
```

➡ Click **Go**

STEP 12: Create `voids`

```
CREATE TABLE voids (
    void_id INT AUTO_INCREMENT PRIMARY KEY,
    order_item_id INT,
    reason VARCHAR(255),
    FOREIGN KEY (order_item_id) REFERENCES order_items(order_item_id)
);
```

➡ Click **Go**

PART 6: Insert Sample Data (VERY IMPORTANT ORDER)

STEP 13: Insert Waiters

```
INSERT INTO waiters (waiter_name) VALUES
('Juan Dela Cruz'),
('Maria Santos');
```

➡ Click **Go**

STEP 14: Insert Restaurant Tables

```
INSERT INTO restaurant_tables (capacity) VALUES
```

(4),
(6);

➡ Click **Go**

STEP 15: Insert Printers

INSERT INTO printers (printer_name) VALUES
('Cold Item Printer'),
('Hot Item Printer'),
('Bar Printer');

➡ Click **Go**

STEP 16: Insert Menu Items

INSERT INTO menu_items (item_name, price, printer_id, in_stock) VALUES
('Caesar Salad', 150.00, 1, TRUE),
('Hot Sandwich', 200.00, 2, TRUE),
('Iced Tea', 80.00, 3, TRUE);

➡ Click **Go**

STEP 17: Insert Orders

INSERT INTO orders (waiter_id, table_id) VALUES
(1, 1),
(2, 2);

➡ Click **Go**

STEP 18: Insert Order Items

```
INSERT INTO order_items (order_id, item_id, quantity) VALUES
(1, 1, 1),
(1, 3, 2),
(2, 2, 1);
```

➡ Click **Go**

STEP 19: Insert Voided Item

```
INSERT INTO voids (order_item_id, reason) VALUES
(3, 'Customer changed mind');
```

➡ Click **Go**

PART 7: Verify That It Worked

STEP 20: Check Tables

1. Click **restaurant_system**
2. Click any table (e.g., `menu_items`)
3. Click **Browse**

You should see the data.
