**Anjapar Restaurant Management System**

**📁 *Database Project Description Summary***

**🧩 Overview**

The **Anjapar Restaurant** database simulates a restaurant management system built using **SQL**. It manages **branches**, **customers**, **menus**, **orders**, **tables**, and **reservations** using a structured schema under the database name **Anjapar Restaurant**.

**🎯 Objective**

To build a comprehensive, scalable SQL-based restaurant management system that supports:

* Customer tracking
* Order & reservation management
* Efficient data querying using:  
  **SELECT, INSERT, UPDATE, ALTER, LIKE, JOINs, SUBQUERIES, STORED PROCEDURES**

**🏗️ Database Setup**

Design a new database schema called **Anjapar Restaurant** including:

* Table creation
* Relationships (foreign keys)
* Sample data for testing

**🔧 Features**

**🔹 1. Branch Management**

* Add / view / update branch information
* Track **menu items**, **number of tables**, and **reservations** per branch
* Designed to support **easy scalability** for new branches

**🔹 2. Customer Management**

* Add / manage **customer info** (name, phone)
* Link customers to **orders** and **reservations**
* Generate queries to find **regular or frequent customers**

**🔹 3. Order Management**

* Create and track **customer orders**
* Store order details like **items** and **date**
* Link orders to both **branches** and **customers**
* Generate reports for **total orders per branch or customer**

**🔹 4. Reservation Management**

* Allow customers to **reserve tables** at specific branches
* Store **reservation date**, **table number**, and **branch info**
* Implement validation to **prevent double-booking**

**🔹 5. Menu Management**

* Maintain a **menu list per branch**
* Optionally normalize into a **separate Menu table** for future scalability
* View available menu items by branch

**🔹 6. Search & Filter (Wildcard)**

* Use SQL **LIKE** operator for flexible searches
* *Example:* Search for customers whose names **start with ‘S’**

**🔹 7. Data Relationships (JOINs)**

Use INNER JOIN to retrieve combined data across tables:

* Orders with **customer** and **branch**
* Reservations with **customer** and **branch**

**🔹 8. Reports & Queries**

Use **subqueries** and **GROUP BY** to generate insights:

* Top branches by **number of orders**
* Customers with **multiple reservations**
* Generate **daily, weekly, or monthly** summaries

**🔹 9. Stored Procedures**

Automate common queries using stored procedures:

* Show all orders from a **specific branch**
* List customers with **reservations**

**🔹 10. Scalability**

The schema supports future extensions, including:

* Additional **branches** or **menu items**
* New features: **payments**, **employee tracking**, **delivery services**, etc.

**📊 Queries & Analysis**

| **🔍 Query Type** | **💡 Use Case** |
| --- | --- |
| SELECT | View full data from any table |
| JOIN | Combine customer, order, and branch data |
| WILDCARD (LIKE) | Search customers by partial name |
| GROUP BY | Count orders per branch or customer |
| SUBQUERY | Filter based on related table data (e.g. Chennai) |
| STORED PROCEDURE | Reusable logic for frequent queries |

**🧪 Example Queries**

1. ✅ View All Customers
2. ✅ View All Branches
3. ✅ View All Orders
4. ✅ View All Reservations
5. 🔍 **Wildcard Search** – Customers starting with ‘S’
6. 🔗 **JOIN Orders** with Customer and Branch
7. 🔗 **JOIN Reservations** with Customer and Branch
8. 📆 View Orders on a Specific Date
9. 🔢 Count of Orders per Branch
10. 👥 **Most Active Customer** (by order count)
11. 📍 **Subquery** – Find Customers who Ordered from **Chennai Branch**