**Capstone Project Title:**

**"Smart City Services – A Data-Driven Municipal Management System"**

**🧠 Objective:**

Students will design and implement a SQL-based database system for a fictional SmartCity that tracks multiple departments like Utilities, Citizens, Services, Billing, and Complaints. This project will cover **every major SQL topic**, including:

* Database design and normalization
* DDL (Data Definition Language)
* DML (Data Manipulation Language)
* Joins (INNER, LEFT, RIGHT, FULL)
* Aggregate functions and GROUP BY
* Subqueries and CTEs
* Views
* Indexes
* Stored Procedures
* Triggers
* Transactions and ACID properties
* User Roles and Permissions
* Optimization Techniques

**📦 Project Requirements:**

**1. Database Schema Design**

Design a normalized relational schema (3NF or BCNF) for the following entities:

* **Citizens** (ID, Name, Address, Contact Info, Registration Date)
* **Departments** (ID, Name, Head, Budget)
* **Services** (ID, Name, Description, DepartmentID, Cost)
* **Requests** (RequestID, CitizenID, ServiceID, DateRequested, Status)
* **Complaints** (ComplaintID, CitizenID, ServiceID, Date, Description, Status)
* **Billing** (BillID, CitizenID, ServiceID, Amount, DueDate, Status)
* **Payments** (PaymentID, BillID, PaymentDate, Amount, Method)

**📐 Key Deliverables (with SQL Focus)**

**📌 1. Schema Creation (DDL)**

* Write CREATE TABLE statements for all entities with appropriate data types and constraints (PK, FK, UNIQUE, CHECK, NOT NULL).

**📌 2. Data Insertion (DML)**

* Insert sample data (at least 100 rows distributed across tables).
* Use INSERT, UPDATE, DELETE.

**📌 3. Data Retrieval & Analysis**

* Write queries to answer questions like:
  + Top 5 most requested services
  + Total revenue generated by each department
  + Citizens with unpaid bills over 60 days
  + Number of complaints per department
  + Average resolution time for services

**📌 4. Joins**

* Use INNER, LEFT, RIGHT, and FULL OUTER JOIN to combine data from multiple tables.

**📌 5. Aggregations & Grouping**

* Use SUM(), AVG(), COUNT(), MAX(), MIN() with GROUP BY and HAVING.

**📌 6. Subqueries and CTEs**

* Nested queries for complex analysis (e.g., citizens who’ve made more than 5 complaints).
* Use WITH clause to build readable queries.

**📌 7. Views**

* Create reusable views such as:
  + Active citizens with billing info
  + Services pending assignment

**📌 8. Stored Procedures**

* Write stored procedures for:
  + Generating monthly billing reports
  + Assigning complaints to the right department

**📌 9. Triggers**

* Create a trigger to:
  + Auto-update complaint status when resolved
  + Log changes to the billing table

**📌 10. Transactions**

* Demonstrate transaction control:
  + BEGIN, COMMIT, ROLLBACK
  + Example: payment and bill status update in one transaction

**📌 11. Indexing & Optimization**

* Identify frequently used queries and create indexes
* Use EXPLAIN PLAN or similar tools (based on DBMS) to demonstrate optimization

**📌 12. Roles & Permissions**

* Create users with specific roles:
  + Admin (full access)
  + Department Head (read-only to billing and complaints)
  + Citizen Portal (limited access to their own data)

**✅ Expected Outcomes**

By completing this project, students will:

* Understand and apply full-stack SQL knowledge
* Be able to design and implement normalized databases
* Write optimized and secure SQL code
* Simulate real-world enterprise use cases
* Demonstrate full CRUD operations, analysis, and reporting using SQL

**🛠 Optional Add-ons (for advanced learners)**

* **Connect database with a front-end** (basic UI using Python Flask or Power BI for reporting)
* **Import/Export data** using CSV or JSON
* **Scheduled backups or reporting using events (MySQL) or jobs (SQL Server)**