Project Title	**Healthcare Analytics with SQL**
Skills take away From This Project	**Proficiency in SQL Joins (Inner, Left, Right, Full Outer).** **Advanced usage of Aggregate Functions (e.g., COUNT, SUM).** **Expertise in Window Functions (e.g., RANK, DENSE_RANK).** **Working with Subqueries and Conditional Logic.** **Understanding of Date and Time Functions for analyzing temporal data.** **Experience with complex data analysis and reporting in a healthcare context**
Domain	**Healthcare Analytics**

Problem Statement:

The project aims to analyze healthcare data, focusing on extracting meaningful insights about patients, doctors, appointments, diagnoses, and treatments using advanced SQL techniques. Learners will apply SQL operations like joins, subqueries, window functions, and more to analyze healthcare metrics.

Business Use Cases:

Patient Management: Analyzing patient demographics, their medical histories, and appointment trends.

Doctor Performance Evaluation: Measuring doctor efficiency by tracking diagnoses and treatment frequency.

Appointment Scheduling & Completion: Understanding appointment trends, cancellations, and completions.

Medication Analysis: Identifying the most common medications prescribed to patients based on diagnoses.

Revenue and Billing Analysis: Linking appointment data with billing information to understand revenue generation.

Approach:

Inner and Equi Joins

Task: Write a query to fetch details of all completed appointments, including the patient's name, doctor's name, and specialization.
Expected Learning: Demonstrates understanding of Inner Joins and filtering conditions.

Left Join with Null Handling

Task: Retrieve all patients who have never had an appointment. Include their name, contact details, and address in the output.
Expected Learning: Use of Left Joins and handling NULL values.

Right Join and Aggregate Functions

Task: Find the total number of diagnoses for each doctor, including doctors who haven't diagnosed any patients. Display the doctor's name, specialization, and total

diagnoses. Expected Learning: Utilization of Right Joins with aggregate functions like COUNT().

Full Join for Overlapping Data

Task: Write a query to identify mismatches between the appointments and diagnoses tables. Include all appointments and diagnoses with their corresponding patient and doctor details.

Expected Learning: Handling Full Joins for comparing data across multiple tables.

Window Functions (Ranking and Aggregation)

Task: For each doctor, rank their patients based on the number of appointments in descending order.

Expected Learning: Application of Ranking Functions such as RANK() or DENSE_RANK().

Conditional Expressions

Task: Write a query to categorize patients by age group (e.g., 18-30, 31-50, 51+). Count the number of patients in each age group.

Expected Learning: Using CASE statements for conditional logic.

Numeric and String Functions

Task: Retrieve a list of patients whose contact numbers end with "1234" and display their names in uppercase.

Expected Learning: Use of string functions like UPPER() and LIKE.

Subqueries for Filtering

Task: Find patients who have only been prescribed "Insulin" in any of their diagnoses.

Expected Learning: Writing Subqueries for advanced filtering.

Date and Time Functions

Task: Calculate the average duration (in days) for which medications are prescribed for each diagnosis.

Expected Learning: Working with date functions like DATEDIFF().

Complex Joins and Aggregation

Task: Write a query to identify the doctor who has attended the most unique patients. Include the doctor's name, specialization, and the count of unique patients.

Expected Learning: Combining Joins, Grouping, and COUNT(DISTINCT).

Results:

- Extract insights about completed appointments and patient histories.
- Analyze the performance of doctors in terms of diagnoses.
- Identify patients who have never had appointments.
- Categorize patients into age groups and rank doctors based on patient visits.
- Handle missing data and use subqueries to filter data efficiently.

Project Evaluation metrics:

Accuracy of SQL Queries: Correctness in terms of logic and SQL syntax.

Data Insights: Ability to extract meaningful information from the dataset.

Efficiency: Optimization of queries to reduce execution time.

Complexity: Proper use of **window functions**, **joins**, and **subqueries**.

Code Documentation: Clear and concise documentation of the code.

Technical Tags:

SQL

Joins (Inner, Left, Right, Full Outer)

Aggregate Functions

Window Functions

Subqueries

Data Analysis

Healthcare Analytics

Data Set:

Healthcare-Dataset

1. **Patients Table**

   Contains details of patients, such as demographics, contact information, and medical history.

2. **Doctors Table**

   Includes information about doctors, their specialties, and contact details.

3. **Appointments Table**

   Tracks appointment schedules, patient-doctor assignments, and visit dates.

4. **Treatments Table**

   Records details of treatments provided to patients, including procedures, costs,

and related patient- doctor interactions.

5. **Medications Table**

   Contains data on prescribed medications, their dosage, and related patient IDs.

Data Set Explanation:

1. Patients Dataset

Details: Information about patients, such as their names, age, gender, and contact details.

Data Types:

- **PatientID**: Number (INT) - Unique identifier for each patient.
- **Name**: Text (VARCHAR) - Patient's full name.
- **Age**: Number (INT) - Age of the patient.
- **Gender**: Text (VARCHAR) - Gender of the patient.
- **ContactNumber**: Text (VARCHAR) - Patient's contact number.
- **Address**: Text (VARCHAR) - Address of the patient.

2. Doctors Dataset

Details: Information about doctors, including their names, contact details, and specialization.

Data Types:

- **DoctorID**: Number (INT) - Unique identifier for each doctor.
- **Name**: Text (VARCHAR) - Doctor's full name.
- **Specialization**: Text (VARCHAR) - Area of expertise (e.g., Cardiologist, Neurologist).

- **ContactNumber**: Text (VARCHAR) - Doctor's contact number.

3. Appointments Dataset

Details: Information about appointments, including patient-doctor interactions, date, and status.

Data Types:

- **AppointmentID**: Number (INT) - Unique identifier for each appointment.
- **PatientID**: Number (INT) - Links to the Patients dataset.
- **DoctorID**: Number (INT) - Links to the Doctors dataset.
- **AppointmentDate**: Date (DATE) - Date and time of the appointment.
- **Status**: Text (VARCHAR) - Status of the appointment (e.g., Completed, Cancelled, Pending).

4. Diagnoses Dataset

Details: Diagnoses provided during patient appointments, along with prescribed medications.

Data Types:

- **DiagnosisID**: Number (INT) - Unique identifier for each diagnosis.
- **AppointmentID**: Number (INT) - Links to the Appointments dataset.
- **Diagnosis**: Text (VARCHAR) - Medical condition diagnosed.
- **Medication**: Text (VARCHAR) - Medication prescribed for the condition.

5. Medications Dataset

Details: Details about medications prescribed to patients, including dosages.

Data Types:

- **MedicationID**: Number (INT) - Unique identifier for each medication.
- **DiagnosisID**: Number (INT) - Links to the Diagnoses dataset.

- **MedicationName**: Text (VARCHAR) - Name of the medication.
- **Dosage**: Text (VARCHAR) - Recommended dosage.

Project Deliverables:

1. **SQL Scripts:** Queries and commands used in the project.
2. **ER Diagram:** Visual representation of the database structure.
3. **Reports:** Output for business questions.
4. **Documentation:** Explanation of approach, assumptions, and findings.

.Project Guidelines:

Coding Standards: Write clean and readable SQL code. Use meaningful variable and column names.

Documentation: Provide comments for each SQL query explaining its purpose and the expected output.

Error Handling: Ensure that queries handle NULL values and missing data correctly.

Optimization: Focus on writing optimized queries that can scale for larger datasets.

Timeline:

The project must be completed and submitted **within 7 days from the assigned date**.

Approval Workflow

Created By:	Verified By:	Approved By:
M S Gaurav Kumar	Poornam Annamalai	Poornam Annamalai