Building Blocks for Healthcare Data

A technical portfolio showcasing the design and implementation of a scalable Healthcare Database Management System (DBMS) using MySQL and SQL. This presentation is tailored for IT professionals and hiring managers in the healthcare sector.

PROJECT OVERVIEW

Core Competencies: SQL & Database Design

Normalized Database Implementation

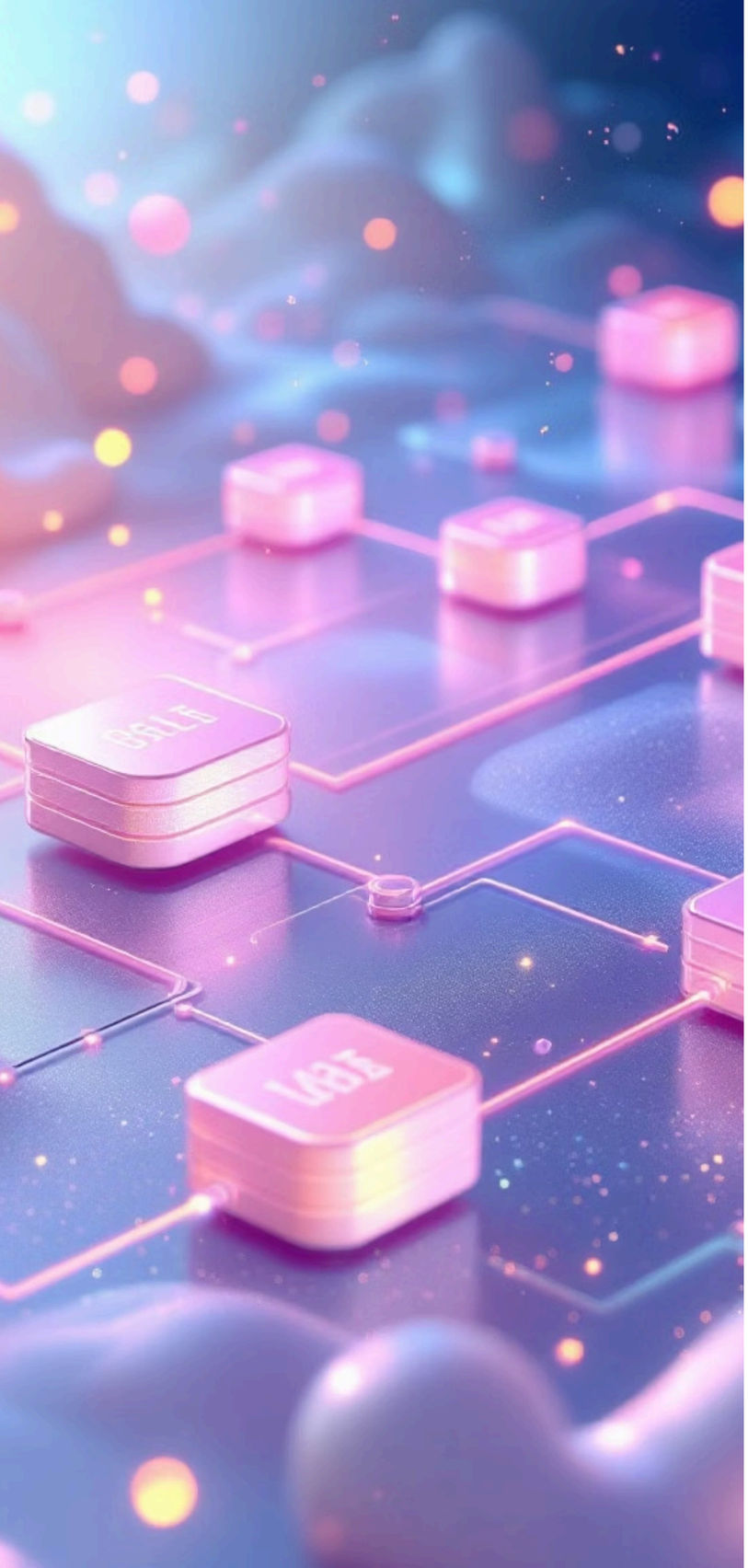
Designed and implemented a relational healthcare database using MySQL, ensuring third normal form (3NF) for data integrity and minimal redundancy across key tables.

Advanced Query Development

Authored complex SQL queries for efficient data retrieval, comprehensive reporting, and crucial analytical insights required for operational decision-making in a clinical setting.

Healthcare Data Modeling

Modeled essential clinical entities: Patients, Doctors, Appointments, Medications, and Prescriptions, establishing robust relationships via foreign keys for transaction reliability.



The Relational Schema: A Foundation for Health Data

The database structure ensures a logical separation of concerns, maintaining data quality and supporting complex analytical requirements. Below is the initial schema design using standard SQL conventions.

```
CREATE DATABASE HealthcareDB;
USE HealthcareDB;

-- Patient table
CREATE TABLE Patients (
 patient_id INT PRIMARY KEY,
 name VARCHAR(100),
 age INT,
 gender VARCHAR(10),
 contact VARCHAR(15)
);

-- Doctor table
CREATE TABLE Doctors (
 doctor_id INT PRIMARY KEY,
 name VARCHAR(100),
 specialization VARCHAR(50),
 contact VARCHAR(15)
);
```

Deep Dive: Appointments and Prescriptions

The core of the clinical data revolves around the relationship between appointments and prescriptions, linked by foreign keys for referential integrity.

Appointments Table

Links patients and doctors, tracking the diagnosis for each visit.

Key Fields: appointment_id, patient_id (FK), doctor_id (FK), appointment_date, diagnosis.

Medications Table

Catalog of available drugs with detailed descriptions.

Key Fields: medication_id, name, description.

Prescriptions Table

Details the specific medication and dosage tied to an appointment.

Key Fields: prescription_id, appointment_id (FK), medication_id (FK), dosage.

Sample Data Insertion for Testing

Realistic sample data was inserted to validate the schema design and test the performance of the analytical queries.

```
INSERT INTO Patients VALUES
(1, 'Ali Khan', 35, 'Male', '9001234567'),
(2, 'Sara Sheikh', 28, 'Female', '9012345678');

INSERT INTO Doctors VALUES
(1, 'Dr. Arjun Rao', 'Cardiology', '9123456789'),
(2, 'Dr. Meera Shah', 'Dermatology', '9234567890');

INSERT INTO Appointments VALUES
(1, 1, 1, '2025-06-20', 'High BP'),
(2, 2, 2, '2025-06-25', 'Skin Allergy');

INSERT INTO Medications VALUES
(1, 'Atenolol', 'Used for high blood pressure'),
(2, 'Cetirizine', 'Used for allergies');

INSERT INTO Prescriptions VALUES
(1, 1, 1, '50mg daily'),
(2, 2, 2, '10mg daily');
```

This sample set represents two distinct clinical scenarios: a cardiovascular check-up and an allergy consultation, proving the system's ability to handle diverse medical records.

QUERY DEVELOPMENT

Operational Reporting with Standard SQL

1

List All Patients & Doctors

```
SELECT * FROM Patients;
SELECT name, specialization FROM Doctors;
```

2

Appointment Details (JOIN Query)

Retrieving full appointment context, linking patient and doctor names via multi-table JOINs.

```
SELECT a.appointment_id, p.name AS patient_name, d.name AS doctor_name, a.appointment_date FROM Appointments a JOIN Patients p ON a.patient_id = p.patient_id JOIN Doctors d ON a.doctor_id = d.doctor_id;
```

3

Basic Demographic Counts

Quickly summarizing patient population metrics (e.g., total count, male patients).

```
SELECT COUNT(*) AS total_patients FROM Patients;
```

Advanced Analytical Queries for Clinical Insight

Leveraging aggregation, grouping, and ordering to extract critical business intelligence from the clinical data.

Age-Based Filtering

Identifying specific demographics for targeted health programs or screening. Queries included finding the oldest and youngest patients, as well as filtering by age criteria.

Example: `SELECT * FROM Patients WHERE age > 30;`

Service Utilization & Load

Calculating total appointments per doctor to assess workload distribution and resource allocation efficiency.

Example: `SELECT d.name, COUNT(*) AS total_appointments FROM Appointments a JOIN Doctors d ON a.doctor_id = d.doctor_id GROUP BY d.name;`

Clinical Specialization Trends

Determining the most common doctor specialization for strategic hiring and facility planning based on demand.

Example: `SELECT specialization, COUNT(*) FROM Doctors GROUP BY specialization ORDER BY count DESC LIMIT 1;`

Summary: Scalable Solutions for Healthcare IT

This project demonstrates proficiency in building robust, performant, and reliable database structures essential for modern healthcare systems.

> Core competencies include not only database design but also the critical ability to translate operational needs into optimized, analytical SQL queries.

Key Takeaways:

- Proficiency in MySQL for structured clinical data management.
- Expertise in complex SQL JOINs and aggregation functions for reporting.
- Commitment to data integrity and security through robust relational design.