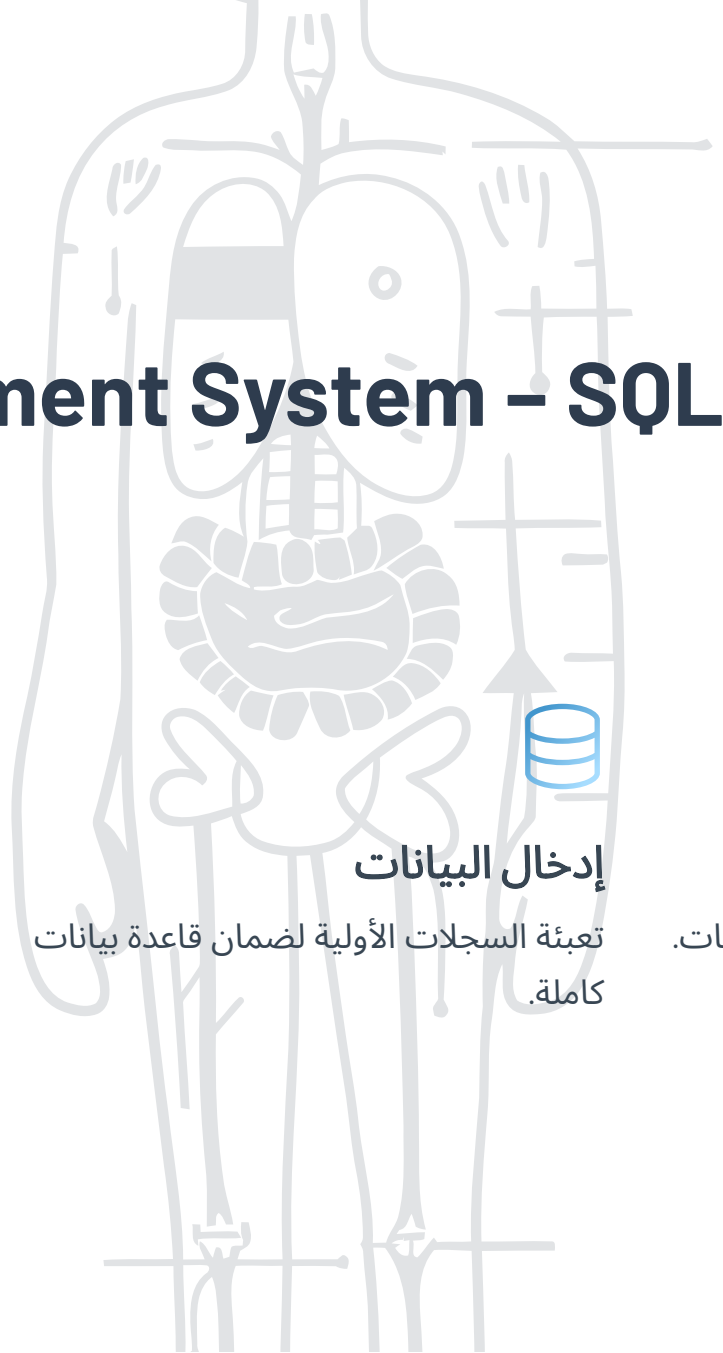
Hospital Management System – SQL Practice Project

Prepared by Mahmoud



تصميم قاعدة البيانات

هيكل بيانات شامل لنظام إدارة المستشفيات.

إدخال البيانات

تعبئة السجلات الأولية لضمان قاعدة بيانات كاملة.

27 استعلام T-SQL

تحليل واسترجاع المعلومات بكفاءة من قاعدة البيانات.

Project Overview

This document presents a comprehensive SQL practice project implementing a full Hospital Management System using Microsoft SQL Server (T-SQL). It includes:

- Database schema design
- Data insertion scripts
- 27 representative T-SQL queries demonstrating key operations

Key Technologies and Concepts

SQL Server Management Studio (SSMS)

T-SQL Programming

Database Design & Normalization

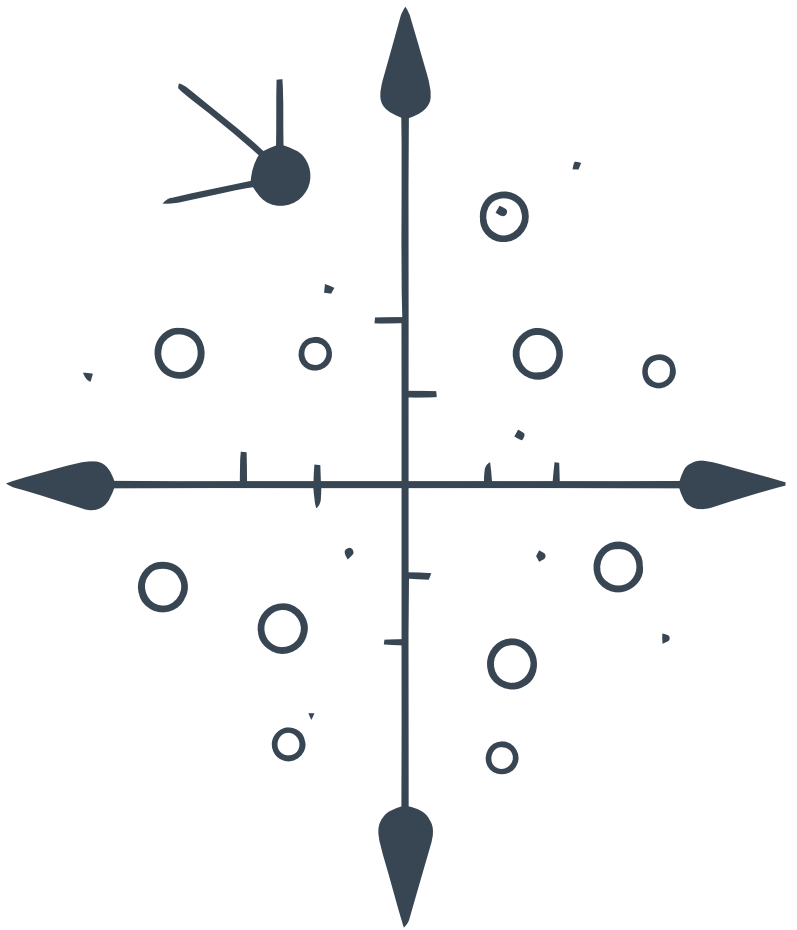
Query Optimization & Performance Tuning

Data Analysis and Reporting

Database Schema (CREATE TABLE Scripts)

The database schema design is a critical component of the Hospital Management System, ensuring data integrity and efficient storage. You can access the detailed CREATE TABLE scripts via the link below:

https://drive.google.com/file/d/1nQdq6UsPAYrUeL-Dz06Qxrr7GKcTSI3o/view?usp=drive_link



Data Insertion

To populate the database with sample data for testing and demonstration, comprehensive data insertion scripts have been prepared. These scripts ensure that the database is ready for query execution and analysis.

https://drive.google.com/file/d/1ouxfiEHbiDxBcZtS1CrrRh_l2jXKAX5J/view?usp=drive_link

27 Representative T-SQL Queries

This section showcases 27 representative T-SQL queries designed to demonstrate key operations within the Hospital Management System. These queries cover a wide range of functionalities, from basic data retrieval to more complex data manipulation and analysis.

The following queries illustrate various aspects of database interaction and management.

Database and Table Renaming

Rename Table

```
;'EXEC sp_rename 'patients', 'clients
```

This command renames the 'patients' table to 'clients', which might be used to align with a broader terminology within the system.

Rename Database

```
ALTER DATABASE [Hospital DB] MODIFY NAME =
;[Medical Center DB]
```

This command renames the entire database from "Hospital DB" to "Medical Center DB", reflecting a potential change in scope or branding.

Data Retrieval and Filtering

Doctors by Department

```
SELECT * FROM doctors WHERE department_id = 'DEPT001';
```

Retrieves all doctors belonging to a specific department.

Appointments by Doctor and Status

```
SELECT * FROM appointments WHERE doctor_id='DOC005' AND status='Completed';
```

Filters appointments for a specific doctor that have been completed.

Bills by Amount and Status

```
SELECT * FROM bills WHERE total_amount > 500 AND payment_status='Pending' AND billing_date >= DATEADD(DAY,-30,GETDATE());
```

Selects bills with a total amount greater than 500, pending status, and billed within the last 30 days.

Medicines by Name Pattern

```
SELECT * FROM medicines WHERE medicine_name LIKE '%cin%';
```

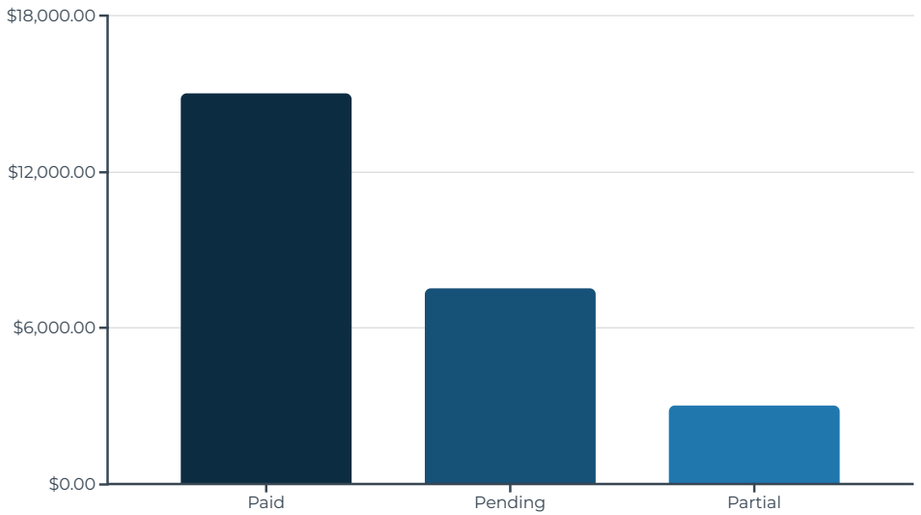
Finds medicines with names containing the substring 'cin'.

Data Manipulation and Aggregation

Total Bill Amount by Payment Status

```
SELECT payment_status, SUM(total_amount) AS total
;FROM bills GROUP BY payment_status
```

Calculates the sum of total amounts for each payment status.



Insert New Appointment

```
INSERT INTO appointments(patient_id, doctor_id,
appointment_date, appointment_time) VALUES
;('PAT025','DOC003','2024-03-10','10:30')
```

Adds a new appointment record to the database.

Update Medicine Stock

```
UPDATE medicines SET stock_quantity = 2500 WHERE
;'medicine_id = 'MED005
```

Updates the stock quantity for a specific medicine.

Count Appointments per Patient

```
SELECT patient_id, COUNT(*) AS total FROM
appointments GROUP BY patient_id HAVING
;COUNT(*)>2
```

Identifies patients with more than two appointments.

Advanced Query Techniques

1 Remove Duplicate Patients

```
WITH cte AS (SELECT *, ROW_NUMBER()
OVER(PARTITION BY
first_name,last_name,date_of_birth ORDER BY
patient_id) AS rn FROM patients) DELETE FROM
cte WHERE rn>1;
```

Uses a Common Table Expression (CTE) to identify and remove duplicate patient records based on name and date of birth.

2 Patients Without Appointments

```
SELECT patient_id FROM patients EXCEPT SELECT
DISTINCT patient_id FROM appointments;
```

Identifies patients who have not made any appointments using the EXCEPT operator.

3 Medicines with Higher Price than a Category

```
SELECT * FROM medicines WHERE unit_price >
ALL (SELECT unit_price FROM medicines WHERE
category='Pain Reliever');
```

Finds medicines whose unit price is greater than all medicines in the 'Pain Reliever' category.

4 Medicines in Prescriptions

```
SELECT * FROM medicines m WHERE EXISTS
(SELECT 1 FROM prescription_medicines p
WHERE p.medicine_id=m.medicine_id);
```

Retrieves medicines that are currently part of any prescription using the EXISTS clause.