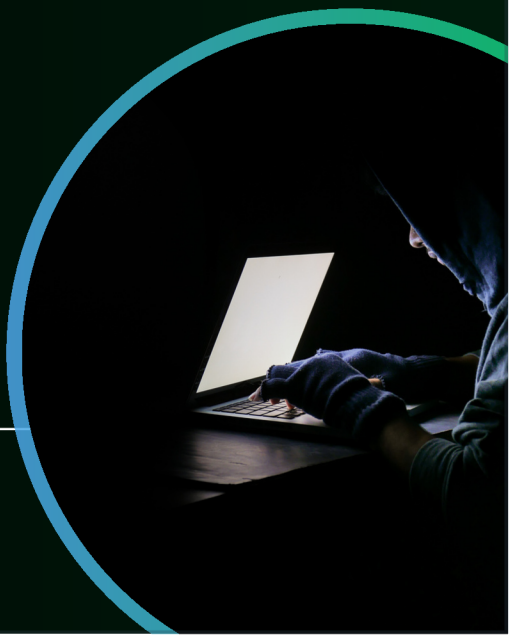
Nile university

Medical Laboratory System Database Design and Implementation

Dr mohammed

sama saeed



Introduction

This project aims to provide a comprehensive database system for a medical laboratory. It oversees critical entities such as laboratory employees, patients, medical tests, test outcomes, and test components. The system improves laboratory operations by arranging test supplies, assuring correct recording of findings, and allowing quick access to patient test data. Additionally, a user-friendly desktop application was created using Python and Tkinter, allowing users to interact with the system without the need for sophisticated instructions.

Programmer Presentation

Entities and Attributes

·Laboratorian with attributes Laboratorian_ID (Primary Key), Name, Phone_Number, and Address.
·Patient with attributes Patient_ID (Primary Key), Name, Phone_Number, Address, Birth_Date, and Job.
·Component with attributes Component_ID (Primary Key), Name, Available_Quantity, and Minimum_Quantity.
·Medical_Test with attributes Test_ID (Primary Key), Name, and Price.
·There is a many-to-many relationship between Medical_Test and Component, represented by the associative entity Test_Component. It has a composite primary key consisting of Test_ID and Component_ID, both of which are foreign keys referencing Medical_Test and Component respectively.
· The Test_Result entity stores test results with attributes Result_ID (Primary Key), Test_ID (Foreign Key referencing Medical_Test), Patient_ID (Foreign Key referencing Patient), Laboratorian_ID (Foreign Key referencing Laboratorian), Date, and Result_Description.

Programmer Presentation

Relational Schema

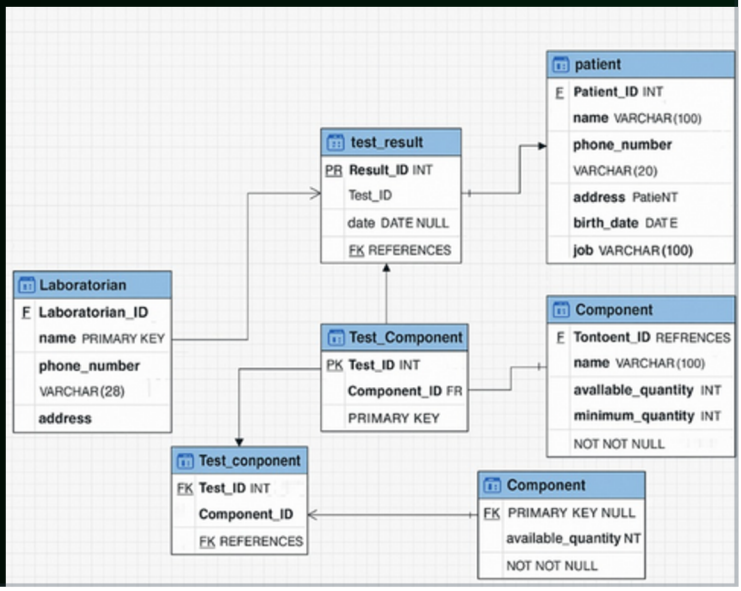
The schema structures the data into tables with well-defined relationships. Each table includes a primary key to uniquely identify records and foreign keys to connect related tables. The Test_Component table employs a composite key to manage the many-to-many relationship between tests and components. This design ensures accurate storage, updating, and retrieval of lab data while maintaining clear and consistent relationships.

Table	Attribute	Keys
Laborato	Laboratorian_ID, Name,	PK: Laboratorian_ID
Patient	Patient_ID, Name, Phc	PK: Patient_ID
Compon	Component_ID, Name, A	PK: Component_ID
Medical_	Test_ID, Name, Price	PK: Test_ID
Test_Cor	Test_ID, Component_ID	PK: (Test_ID, Componen
Test_Res	Result_ID, Test_ID, Date,	PK: Result_ID FK: to relat

ER Diagram

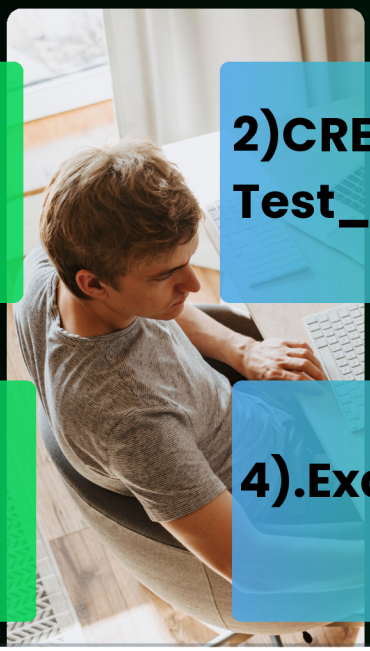
The ER diagram shows relationships in the system.

- A one-to-many relationship between Laboratorian and Test Result
- A one-to-many relationship between Patient and Test Result .
- A many-to-many relationship between Medical Test and Component,
- Test Result requires total participation from Patient and Laboratorian



steps of the squal codes

1)SQL Statements for Table Creation1



2)CREATE TABLE Test_Result

3).Sample Data Insertion

4).Example Queries

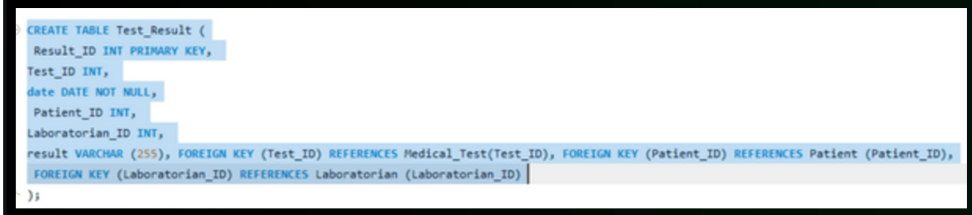
1)SQL Statements for Table Creation

Used SQL to create the tables. I included constraints like PRIMARY KEY, FOREIGN KEY, and NOT NULL to ensure data consistency

```
create database laborATION;
CREATE TABLE Laboratorian (
Laboratorian_ID INT PRIMARY KEY,
name VARCHAR (100) NOT NULL,
 Phone_number VARCHAR (20),
address VARCHAR (255)
 );
 CREATE TABLE Patient (
Patient_ID INT PRIMARY KEY,
name VARCHAR(100) NOT NULL,
 phone_number VARCHAR(20),
address VARCHAR(255),
 birth_date DATE,
 job VARCHAR (100)
);
CREATE TABLE Component (
Component_ID INT PRIMARY KEY,
name VARCHAR(100) NOT NULL,
available_quantity INT NOT NULL,
minimum_quantity INT NOT NULL
);
```

Programmer Presentation

2)CREATE TABLE Test_Result

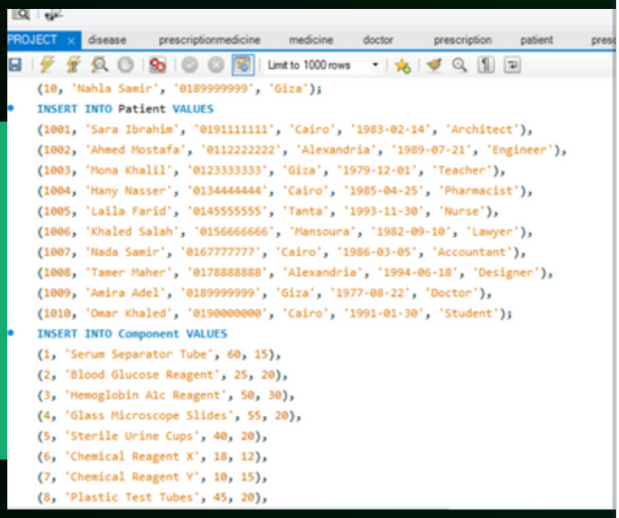


```
CREATE TABLE Test_Result (
 Result_ID INT PRIMARY KEY,
Test_ID INT,
date DATE NOT NULL,
 Patient_ID INT,
Laboratorian_ID INT,
result VARCHAR (255), FOREIGN KEY (Test_ID) REFERENCES Medical_Test(Test_ID), FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID),
 FOREIGN KEY (Laboratorian_ID) REFERENCES Laboratorian (Laboratorian_ID)
);
```

The SQL statements create tables for the medical lab system, with each table having a primary key to uniquely identify records and maintain organization. The Test_Component table uses a composite primary key consisting of Test_ID and Component_ID to represent the many-to-many relationship between medical tests and components. Foreign keys are implemented to connect related tables and ensure

Insert sample data

These SQL insertion commands populate each table with realistic sample data, enabling effective testing of the database. Including at least ten records per table provides sufficient data diversity to thoroughly evaluate queries and system functionality.



Programmer Presentation

4).Example Queries

These queries are used to extract meaningful information from a medical lab database. They identify patients who recently took a specific test, check for low-stock components, and calculate total payments made by individual patients for tests over the past three years. This supports patient tracking, inventory management, and financial reporting.

```sql
SELECT DISTINCT p.name
FROM Patient p
JOIN Test_Result tr ON p.Patient_ID = tr.Patient_ID
JOIN Medical_Test mt ON tr.Test_ID = mt.Test_ID
WHERE mt.name = 'Complete Blood Count'
  AND tr.date >= DATE_SUB(CURDATE(), INTERVAL 1 YEAR);
SELECT name, available_quantity, minimum_quantity
FROM Component
WHERE available_quantity < minimum_quantity;
SELECT p.name, SUM(mt.price) AS total_paid
FROM Patient p
JOIN Test_Result tr ON p.Patient_ID = tr.Patient_ID
JOIN Medical_Test mt ON tr.Test_ID = mt.Test_ID
WHERE p.Patient_ID = 1001
  AND tr.date >= DATE_SUB(CURDATE(), INTERVAL 3 YEAR)
GROUP BY p.Patient_ID;
SELECT p.name, SUM(mt.price) AS total_paid
FROM Patient p
JOIN Test_Result tr ON p.Patient_ID = tr.Patient_ID
JOIN Medical_Test mt ON tr.Test_ID = mt.Test_ID
WHERE p.Patient_ID = 12527
  AND tr.date >= DATE_SUB(CURDATE(), INTERVAL 3 YEAR)
GROUP BY p.Patient_ID;
```

conclusion

In conclusion, designing a well-structured database with clear relationships, proper keys, and realistic sample data is essential for building a reliable and efficient medical lab system. This foundation ensures data integrity, supports accurate testing, and enables smooth operation of the system's functionalities.

Programmer Presentation

Thank You

FOR YOUR ATTENTION

Programmer Presentation