Project: Blood Bank Management System

Mohiuddin

ID: 1280860

Round: 58

Subject: C#

TSP: SCSL

Case Study: Blood Bank Management System

Introduction: The Blood Bank Management System (BBMS) is designed to streamline and manage the various operations of a blood bank efficiently. This system is implemented using a relational database, and the following is a comprehensive case study that covers the database design and the Data Definition Language (DDL) and Data Manipulation Language (DML) statements used in the system.

Database Design:

- ❖ BloodGroups Table:
- ❖ Holds information about different blood groups.
- ❖ Columns: BGroupID, GroupName.
- ❖ Donors Table:
- ❖ Stores details of blood donors.

Columns: DonorID, FirstName, LastName, Age, Gender, BGroupID, ContactNumber, Email, MedicalReport, HemoglobinLevel, BloodPressure, DonarState, DonarAddress, LastDonationDate.

Patients Table:

Contains information about patients needing blood.

Columns: PatientID, FirstName, LastName, Age, Gender, BGroupID, Disease, ContactNumber, Email, PatientState, PatientAddress, MedicalHistory, HemoglobinLevel, BloodPressure.

BloodBanks Table:

Stores data about different blood banks.

Columns: BloodBankID, BloodBankName, BloodBankAddress, BloodBankContact.

BloodStocks Table:

Manages the stock of blood available in the blood bank.

Columns: StockID, BGroupID, QuantityInMilliliters, ExpiryDate, DonorID.

CrossMatch Table:

Records cross-matching information between donors and patients.

Columns: CrossMatchID, PatientID, DonorID, StockID, CrossMatchResult, CrossMatchDate, Remarks.

BloodDonations Table:

Tracks blood donations made by donors.

Columns: DonationID, DonorID, PatientID, DonationDateTime, BGroupID, QuantityInMilliliters.

DDL Statements:

Defines the structure of the database and its tables.

Example: CREATE TABLE BloodGroups (BGroupID INT, GroupName VARCHAR(5));

DML Statements:

Manages the data within the tables.

Example: INSERT INTO BloodGroups(BGroupID, GroupName) VALUES (101, 'A+'), (102, 'A-'), ...

Use Cases:

Donor Registration:

New donors are added to the system using the Donors table.

INSERT INTO Donors ...

Blood Donation:

Records blood donations made by donors.

INSERT INTO BloodDonations ...

Patient Admission:

Adds patient details to the system when admitted.

INSERT INTO Patients ...

Cross-Matching:

Cross-matching results are stored in the CrossMatch table.

INSERT INTO CrossMatch ...

Here is a bulleted list summarizing the SQL DDL statements for creating a **Blood Bank Management System**:

DDL Statements Working Titles:

- Create Database
- Create BloodGroups Table
- Create Donors Table
- Alter Donors Table
- Create Patients, BloodBanks, BloodStocks, CrossMatch, and BloodDonations Tables
- Alter CrossMatch Table
- Delete a Column from CrossMatch Table
- Delete a Table
- Add Non-Clustered Index on BloodDonations Table
- Create Views
- Create Stored Procedures
- Create Triggers
- Create Functions

BloodGroups Table

- ❖ BGroupID (Primary Key)
- ❖ GroupName

Donors Table

- ❖ DonorID (Primary Key)
- ❖ FirstName
- ❖ LastName
- ❖ Age
- ❖ Gender
- ❖ BGroupID (Foreign Key)
- ❖ ContactNumber
- ❖ Email
- ❖ MedicalReport
- ❖ HemoglobinLevel
- ❖ BloodPressure
- ❖ DonarState
- ❖ DonarAddress
- ❖ LastDonationDate

Patients Table

- ❖ PatientID (Primary Key)
- ❖ FirstName
- ❖ LastName
- ❖ Age

- Gender
- BGroupID (Foreign Key)
- Disease
- ContactNumber
- Email
- PatientState
- PatientAddress
- MedicalHistory
- HemoglobinLevel
- BloodPressure

BloodBanks Table

- BloodBankID (Primary Key)
- BloodBankName
- BloodBankAddress
- BloodBankContact
- BloodStocks Table Columns:
- StockID (Primary Key)
- BGroupID (Foreign Key)
- QuantityInMilliliters
- ExpiryDate
- DonorID (Foreign Key)

CrossMatch Table

- CrossMatchID (Primary Key)
- PatientID (Foreign Key)

- DonorID (Foreign Key)
- StockID (Foreign Key)
- CrossMatchResult
- CrossMatchDate
- Remarks
- CrossMatchCost

BloodDonations Table

- DonationID (Primary Key)
- DonorID (Foreign Key)
- PatientID (Foreign Key)
- DonationDateTime
- BGroupID (Foreign Key)
- QuantityInMilliliters

DonorsArchive Table

- DonorID (Primary Key)
- FirstName
- LastName
- Age
- Gender
- BGroupID (Foreign Key)
- ContactNumber
- Email
- MedicalReport
- HemoglobinLevel

- BloodPressure
- DonarState
- DonarAddress
- LastDonationDate

Views:

- DonorsView
- AvgCrossMatch
- EncryptedView

Stored Procedures:

- GetDonorsByState
- spDonorsDetailsByState
- InsertUpdateDeleteOutput

Triggers:

- **trDonorArchiveAfterDeletedFromDonors**
- **InsteadOfTrigger_AgbCrossMatch**

Functions:

- Scalar Function: fnDonorsCountByState
- Table-Valued Function: fnGetDonorsByState
- Multi-Statement Table-Valued Function: fnGetDonorsAndBloodStocksByState

Here are the bulleted points for the DML statements:

- Insert Data into BloodGroups Table:
- INSERT INTO BloodGroups(BGroupID, GroupName) VALUES (101, 'A+'), (102, 'A-'), ...
- Insert Data into Donors Table:
- INSERT INTO Donors (DonorID, FirstName, LastName, Age, Gender, BGroupID, ContactNumber, Email, MedicalReport, HemoglobinLevel, BloodPressure, DonarState, DonarAddress) VALUES ...
- Insert Data into Patients Table:
- INSERT INTO Patients (PatientID, FirstName, LastName, Age, Gender, BGroupID, Disease, ContactNumber, Email, PatientState, PatientAddress, MedicalHistory, HemoglobinLevel, BloodPressure) VALUES ...
- Insert Data into BloodBanks Table:
- INSERT INTO BloodBanks (BloodBankID, BloodBankName, BloodBankAddress, BloodBankContact) VALUES ...
- Insert Data into BloodStocks Table:
- INSERT INTO BloodStocks (StockID, BGroupID, QuantityInMilliliters, ExpiryDate, DonorID) VALUES ...
- Insert Data into CrossMatch Table:
- INSERT INTO CrossMatch (CrossMatchID, PatientID, DonorID, StockID, CrossMatchResult, CrossMatchDate, Remarks) VALUES ...
- Insert Data into BloodDonations Table:
- INSERT INTO BloodDonations (DonationID, DonorID, PatientID, DonationDateTime, BGroupID, QuantityInMilliliters) VALUES ...
- Retrive Data from BloodGroups Table

- Retrive Data from Donors Table
- Retrive Data from Patients Table
- Retrive Data from BloodStocks Table
- Retrive Data from BloodBanks Table
- Retrive Data from CrossMatch Table
- Retrive Data from BloodDonations Table
- Retrive Only Male Donors
- Retrive all Donors Whoes Blood Group A+/A- or any
- Retrive Donors by his/her state
- Format string data using literal values from Donors table
- Uses of LEFT Function
- Calculate Blood Donation Days
- Find out a Patient by his/her id
- Retrive last 3 Donors
- Uses of BETWEEN keyword JOINING TWO TABLES
- Uses of WilCard JOIN WITH CORRELATION NAMES
- user of group by and rollup clause
- Show Donors from 4 to 8 / USES OF OFFSET AND FETCH NEXT
- justify nonclustered index
- justify updatable view
- justify read only view
- justify encrypted view
- check stored procedure
- Stored procedure by output parameter
- --Check stored procedure for insert / update / delete (DML Statement)

        - Select functionality

- INSERT functionality
- UPDATE functionality
- DELETE functionality
- justify after delete trigger
- execution of scaler function
- Example usage of the table-valued function
- Execution usage of the multi-statement function

Conclusion: The Blood Bank Management System, implemented using SQL DDL and DML statements, provides an organized and efficient way to manage blood bank operations, donor information, and patient records. The use of a relational database ensures data integrity and ease of access for various stakeholders involved in blood bank management.

Thanks and Regards

Mohiuddin

ID: 1280860

Round: 58

Subject: C#

TSP: SCSL