ACID Documentation

Hospital, Pharmacy, and Blood Bank Management System

Introduction

This document describes how ACID (Atomicity, Consistency, Isolation, and Durability) properties are maintained in the Hospital, Pharmacy, and Blood Bank Management System. This system involves multiple users such as doctors, receptionists, pharmacists, blood managers, and administrators accessing shared data.

Main Entities

- Doctor
- Receptionist
- Patient
- ER Shifts
- OT
- Pharmacist
- Medicine Bank
- Blood Manager
- Donors
- Blood Bank
- Nurses
- Appointment

Management Tables

- OT_Management
- Prescription_Management
- ER_Management
- Blood_Request

Transaction Scenario 1: Prescription Fulfillment

Scenario Description

The doctor prescribes the medicine to the patient. The patient goes to the pharmacist and the pharmacist checks the availability of medicine, gives the medicine and updates the stock of medicine.

Tables Involved:

Doctor, Patient, Prescription_Management, Medicine, Pharmacist

Transaction Steps

- Verify that the patient exists and he has an appointment
- Doctor prescribes him the medicines
- The patient asks the pharmacist for medicine.
- Pharmacist checks the medicine availability and Expiry dates.
- Deduct medicine quantity from the stock.
- Update the status of the prescription to be fulfilled.

ACID Property Mapping

Properties	Description
Automicity	Either all steps complete together (appointment verified, prescription created, stock deducted, statuses updated) or nothing changes. If the medicine is out of stock, expired, or any step fails, the entire transaction rolls back and the database remains as it was before.
Consistency	Medicine stock cannot be negative. Prescription must reference a valid Doctor_ID, Patient_ID, and Pharmacist_ID. Only non expired medicines can be dispensed. Appointment must exist and be in “pending” status before a prescription is created. All FK, NOT NULL, and CHECK constraints are enforced throughout.
Isolation	Two pharmacists cannot simultaneously dispense from the same medicine batch and over draw stock.
Durability	Once the prescription is marked 'completed' and stock is updated, all changes are permanent. Even if the system crashes immediately after the commit, the prescription record, stock deduction, and appointment status update will be recoverable and will not be lost.

REPEATABLE READ: Repeatable Read means if multiple pharmacists try to dispense the same medicine at the same time, the medicine stock will be updated correctly without lost updates or incorrect deductions.

Concurrency Strategy: A locking mechanism(strict two-phase locking) is used on this table so two pharmacists cannot modify the same stock at the same time which prevents lost updates.

Transaction Scenario 2: Blood Donation Processing

Scenario Description

A donor arrives at the hospital to donate blood. The blood manager verifies donor eligibility (minimum 90 days since last donation), conducts health screening, collects blood, updates blood inventory, and records the donation. This transaction ensures donors don't donate too frequently (health safety) and blood inventory is accurately maintained.

Tables Involved: Donors, Blood,Blood_Manager

Transaction Steps:

- Verify donor eligibility by checking Last_Donation date (it must be ≥90 days ago)
- Reserve donor record to prevent duplicate donations at multiple locations
- Reserve Blood inventory entry for donor's blood group
- Increment blood units in Blood table
- Update donor's Last_Donation date to current date
- Record collection date in Blood table for tracking purposes

ACID Property Mapping

Properties	Description
Automicity	Blood deduction and request approval happen together. If there isn't enough blood available, the entire operation is rolled back. No partial changes are saved to the database.
Consistency	Blood quantity cannot become negative. Blood groups must match the donor's blood type. Referential integrity between Blood table and Donors table is maintained. Donors cannot donate too frequently (90 days minimum).
Isolation	The same donor cannot donate at two locations simultaneously. The first transaction verifies eligibility (95 days ≥ 90 days) and updates Last_Donation. The second waits, then sees the donor just donated (0 days), fails eligibility, and cancels.
Durability	Once a blood donation is recorded, all updates to Blood inventory and Donor's Last_Donation are permanently stored by writing changes to disk before confirming the transaction.

SERIALIZABLE: Serializable means if multiple blood managers try to process donations for the same donor simultaneously, the donor record and blood inventory will be updated correctly without double donations or rule violations.

Concurrency Strategy: The system uses row-level locking on the Donor and Blood tables.When a donor’s eligibility is being checked,the record will be locked so no other transaction can change it. It will prevent double donation.

Transaction Scenario 3: Appointment Scheduling

Scenario Description:
A receptionist schedules an appointment for a patient with a doctor.

Tables Involved: Receptionist, Patient, Doctor, Appointment

Transaction Steps:
• Verify patient exists
• Verify doctor availability
• Insert appointment record
• Update appointment status to Pending or Approved accordingly
All steps are executed as a single transaction.

ACID Property Mapping

Property	Description
Atomicity	Either the appointment is fully created, or not at all resulting in no data being stored. If the doctor is not available then the transaction is rolled back.
Consistency	Appointment must reference valid p_ID, d_ID, and r_ID. Database constraints (PK, FK, NOT NULL) ensure consistency.
Isolation	More than one receptionist can not book an appointment for the same doctor/time slot simultaneously.
Durability	Once confirmed, appointment data persists even in case of system crash. Changes are written to permanent storage.

SERIALIZABLE: Serializable means if multiple receptionists try to book the same doctor and time slot at the same time, the appointment table will be updated correctly without double bookings

Concurrency Strategy: The system locks the specific doctor's time slot when an appointment is being created. If two appointments are being made at the same time for same doctor, one is succeeded and the other is rolled back