CARE HOSPITAL DATABASE MANAGEMENT SYSTEM

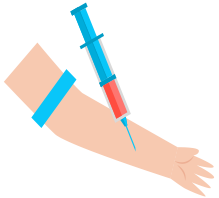
By:
Uzma Naeem

WHO WE ARE

- We are a leading organization in the healthcare sector, dedicated to providing exemplary medical services with a focus on innovation and compassion.
- At Care Hospital, we combine cutting-edge technology with a commitment to patient well-being, ensuring the highest standards of care and excellence in healthcare delivery.

FOCUS

01 Patient care streamlining



02 Seamless staff communication

03 Accurate & secure medical record storage

HIPAA COMPLIANT

SIGNIFICANCE

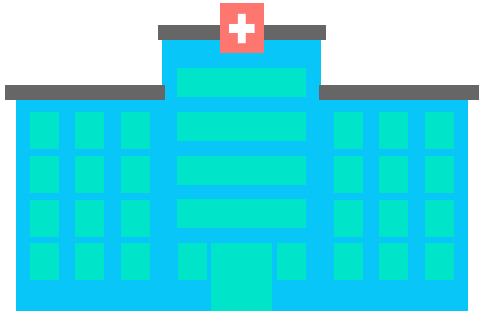
- **Project Significance:** The healthcare sector demands accuracy and efficiency. Our database project addresses these needs by integrating advanced data management technologies that support personalized and efficient medical care.

- **Data Relevance:** Incorporating technology in healthcare is more crucial than ever. This project exemplifies our commitment to leveraging digital solutions to improve healthcare delivery.

ENTITIES & ATTRIBUTES

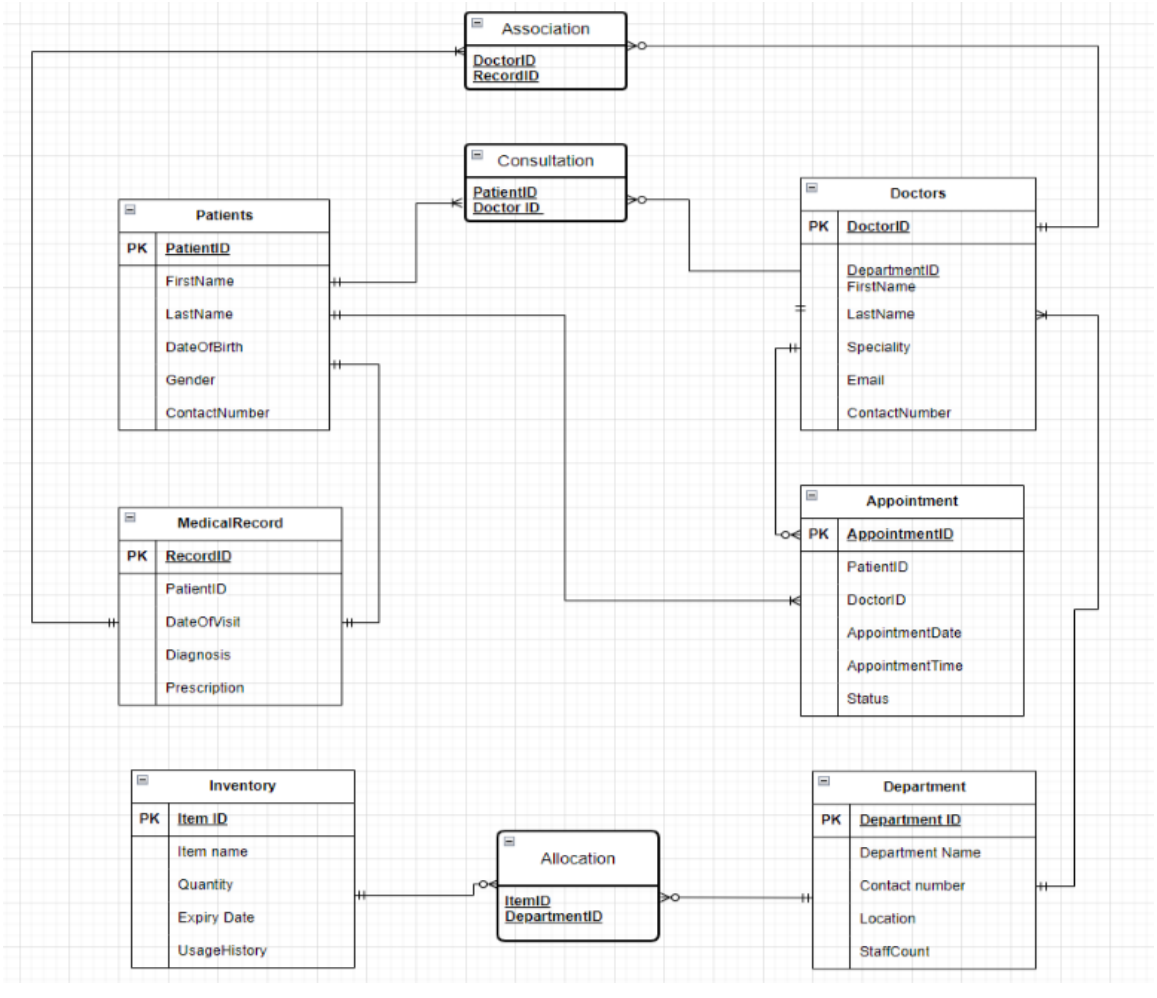
Patient	Doctor	Appointment	Medical Record	Department	Inventory
Patient ID (PK)	Doctor ID (PK)	Appointment ID (PK)	Record ID (PK)	Department ID (PK)	Item ID (PK)
First Name	Department ID	Patient ID	Patient ID	DepartmentName	Item Name
Last Name	First Name	DoctorID	Date Of Visit	Location	Quantity
Date Of Birth	Last Name	AppointmentDate	Diagnosis	Contact Number	Expiry Date
Gender	Specialty	AppointmentTime	Prescription	Staff Count	Usage History
Contact Number	Contact Number	Status			
	Email				

BUSINESS RULES

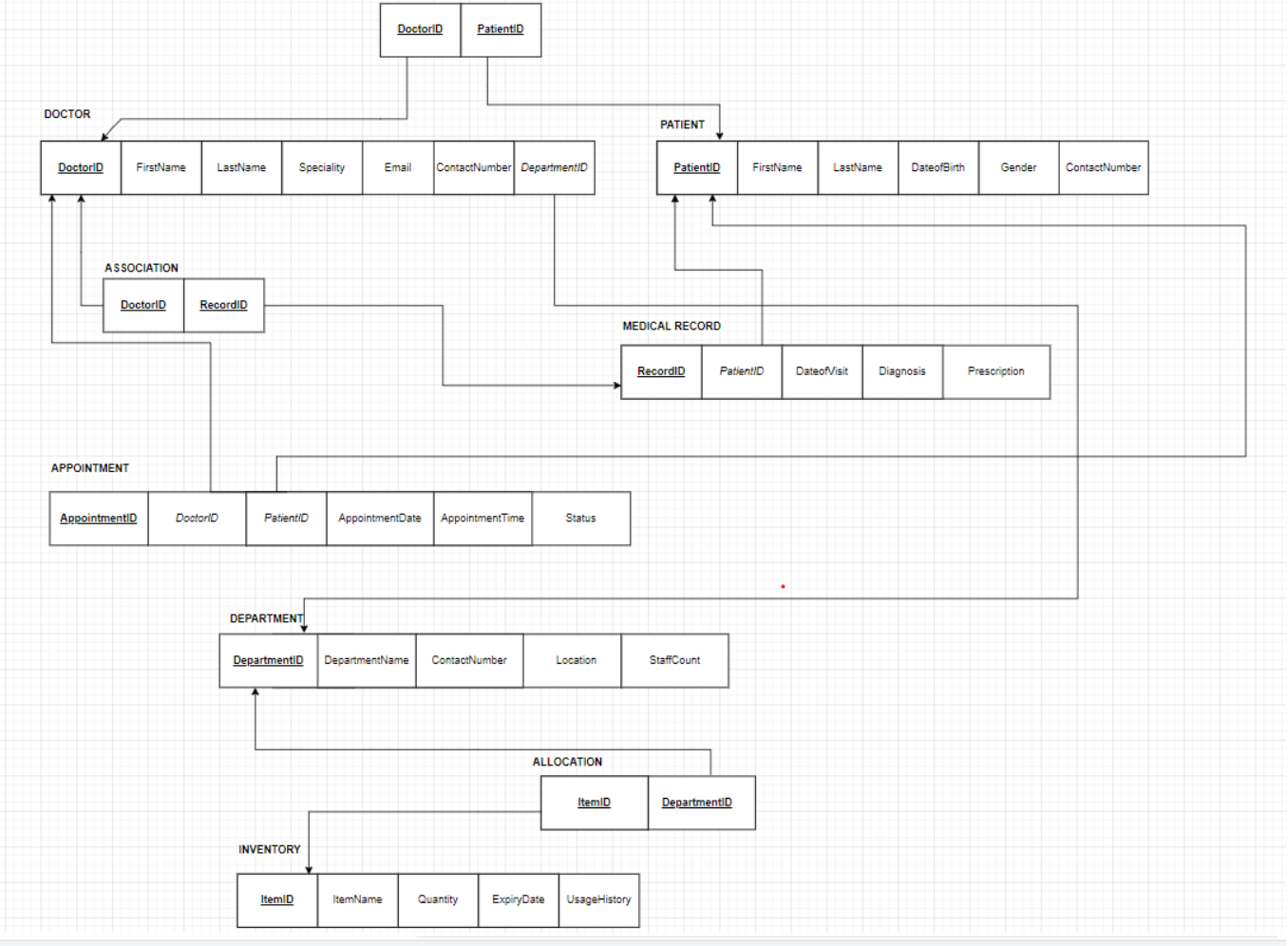


- Each patient **must** be assigned to at least one doctor.
- Each doctor **can** have more than one patient.
- Each patient **must** have one or more appointments.
- Each appointment **must** be associated to one patient and one doctor.
- Each doctor **can** have one or more appointments.
- Each medical record **must** be linked with only one patient.
- Each patient **must** have a single medical record number.
- Each medical record **must** have at least one doctor assigned.
- Each doctor **can** be associated with one or more medical records.
- Each department **must** have more than one doctor.
- Each doctor **must** be assigned to only one department.
- Each department **must** have access to at least one inventory.
- Each inventory **can** be assigned to one or more departments.

ENHANCED ENTITY RELATIONSHIP DIAGRAM



3NF-RELATIONAL MODEL OF CARE HOSPITAL



UNDERSTANDING THE DATA

Data Planning and Implementation:

- To populate the database, data that mirrors real-world hospital operations are simulated.

Simulated Data Usage:

- The simulated data reflects typical patient interactions and healthcare processes, ensuring our system is robust and capable of handling real operational demands.

SQL QUERIES

CREATE & USE A DATABASE :

```sql
-- Create the database
CREATE DATABASE IF NOT EXISTS care_hospital;

-- Use the database
USE care_hospital;
```

CREATE 6 TABLES & INSERT VALUES:

```sql
CREATE TABLE MedicalRecords (
    RecordID INT AUTO_INCREMENT PRIMARY KEY,
    PatientID INT NOT NULL,
    DoctorID INT NOT NULL,
    DateOfVisit DATE NOT NULL,
    Diagnosis TEXT,
    Prescription TEXT,
    CONSTRAINT MedicalRecords_FK1 FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
    CONSTRAINT MedicalRecords_FK2 FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID));
    -- Patients Table (1)
    INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, ContactNumber)
    VALUES
        (101, 'John', 'Smith', '1980-05-15', 'M', '123-456-7890'),
        (102, 'Emily', 'Johnson', '1992-09-20', 'F', '456-789-0123'),
        (103, 'Michael', 'Brown', '1975-03-10', 'M', '789-012-3456'),
        (104, 'Sarah', 'Davis', '1988-11-02', 'F', '234-567-8901'),
        (105, 'David', 'Martinez', '1965-07-25', 'M', '567-890-1234');
```

Result

PatientID	FirstName	LastName	DateOfBirth	Gender	ContactNumber
101	John	Smith	1980-05-15	M	123-456-7890
102	Emily	Johnson	1992-09-20	F	456-789-0123
103	Michael	Brown	1975-03-10	M	789-012-3456
104	Sarah	Davis	1988-11-02	F	234-567-8901
105	David	Martinez	1965-07-25	M	567-890-1234
NULL	NULL	NULL	NULL	NULL	NULL

Alter Patients table to add a column Email

```
ALTER TABLE Patients ADD Column Email VARCHAR(255);
```

Result

PatientID	FirstName	LastName	DateOfBirth	Gender	ContactNumber	Email
101	John	Smith	1980-05-15	M	123-456-7890	NULL
102	Emily	Johnson	1992-09-20	F	456-789-0123	NULL
103	Michael	Brown	1975-03-10	M	789-012-3456	NULL
104	Sarah	Davis	1988-11-02	F	234-567-8901	NULL
105	David	Martinez	1965-07-25	M	567-890-1234	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Update status of a patient's appointment using the appointment ID

```
UPDATE Appointments
SET Status = 'completed'
WHERE AppointmentID = 401;
```

Result

AppointmentID	PatientID	DoctorID	AppointmentDate	AppointmentTime	Status
401	101	301	2024-05-12	10:00:00	scheduled
402	102	302	2024-05-13	11:00:00	scheduled
403	103	303	2024-05-14	12:00:00	completed
404	104	304	2024-05-15	13:00:00	canceled
405	105	305	2024-05-16	14:00:00	scheduled
406	101	301	2024-05-17	15:00:00	completed
NULL	NULL	NULL	NULL	NULL	NULL

Delete an appointment where appointment ID is 404

```
DELETE FROM Appointments
WHERE AppointmentID = 404;
```

BEFORE

AppointmentID	PatientID	DoctorID	AppointmentDate	AppointmentTime	Status
401	101	301	2024-05-12	10:00:00	scheduled
402	102	302	2024-05-13	11:00:00	scheduled
403	103	303	2024-05-14	12:00:00	completed
404	104	304	2024-05-15	13:00:00	canceled
405	105	305	2024-05-16	14:00:00	scheduled
406	101	301	2024-05-17	15:00:00	completed
NULL	NULL	NULL	NULL	NULL	NULL

Drop a column

```
ALTER TABLE Inventory
DROP COLUMN UsageHistory;
```

ItemID	ItemName	Quantity	ExpiryDate	DepartmentID	UsageHistory
601	Bandages	200	2025-12-31	203	Orthopedics
602	Aspirin	500	2023-06-30	201	Cardiology
603	Antibiotics	100	2024-09-30	202	Pediatrics
604	Anti-Allergics	50	2025-03-31	204	Dermatology
605	Apomorphine	300	2024-12-31	205	Neurology
NULL	NULL	NULL	NULL	NULL	NULL

Result

AppointmentID	PatientID	DoctorID	AppointmentDate	AppointmentTime	Status
401	101	301	2024-05-12	10:00:00	scheduled
402	102	302	2024-05-13	11:00:00	scheduled
403	103	303	2024-05-14	12:00:00	completed
405	105	305	2024-05-16	14:00:00	scheduled
406	101	301	2024-05-17	15:00:00	completed
NULL	NULL	NULL	NULL	NULL	NULL

ItemID	ItemName	Quantity	ExpiryDate	DepartmentID
601	Bandages	200	2025-12-31	203
602	Aspirin	500	2023-06-30	201
603	Antibiotics	100	2024-09-30	202
604	Anti-Allergics	50	2025-03-31	204
605	Apomorphine	300	2024-12-31	205
NULL	NULL	NULL	NULL	NULL

Get a count of appointments per doctor, only showing those doctors who have appointments, sort by appointment count in descending order.

```sql
SELECT

CONCAT (Doctors.FirstName, ' ' , Doctors.LastName) AS DoctorName,

COUNT(Appointments.AppointmentID) AS AppointmentCount

FROM Appointments

INNER JOIN Doctors ON Appointments.DoctorID = Doctors.DoctorID

GROUP BY DoctorName

HAVING AppointmentCount > 0

ORDER BY AppointmentCount DESC;
```

Use of Concat, Count, Inner Join, Group By, Having, Order By, Desc

Result

	DoctorName	AppointmentCount
▶	Robert Johnson	2
	Emily Williams	1
	Michael Davis	1
	Sarah Garcia	1
	David Martinez	1

Pull a list of patient's full name who have scheduled appointments, including those who haven't scheduled showing appointment date & time, status, doctor's full name, & department name.

```
SELECT
CONCAT (Patients.FirstName, ' ' , Patients.LastName) AS PatientsName,
 Appointments.AppointmentDate, Appointments.AppointmentTime, Appointments.Status,
CONCAT (Doctors. FirstName, ' ' , Doctors.LastName) AS DoctorsName,
Departments.DepartmentName
FROM Patients
LEFT JOIN Appointments On Patients.PatientID  = Appointments.PatientID
LEFT JOIN Doctors On Appointments.DoctorID = Doctors.DoctorID
LEFT JOIN Departments ON Doctors.DepartmentID = Departments.DepartmentID;
```

Use of Outer Join

Result

PatientsName	AppointmentDate	AppointmentTime	Status	DoctorsName	DepartmentName
John Smith	2024-05-12	10:00:00	scheduled	Robert Johnson	Cardiology
John Smith	2024-05-17	15:00:00	completed	Robert Johnson	Cardiology
Emily Johnson	2024-05-13	11:00:00	scheduled	Emily Williams	Pediatrics
Michael Brown	2024-05-14	12:00:00	completed	Michael Davis	Orthopedics
Sarah Davis	2024-05-15	13:00:00	canceled	Sarah Garcia	Dermatology
David Martinez	2024-05-16	14:00:00	scheduled	David Martinez	Neurology

MULTI TABLE QUERY

Pick a patient who is born after 1980

```
SELECT * FROM Patients

WHERE DateOfBirth > '1980-01-01';
```

Use of Where + greater than condition

Result

	PatientID	FirstName	LastName	DateOfBirth	Gender	ContactNumber
▶	101	John	Smith	1980-05-15	M	123-456-7890
	102	Emily	Johnson	1992-09-20	F	456-789-0123
	104	Sarah	Davis	1988-11-02	F	234-567-8901
*	NULL	NULL	NULL	NULL	NULL	NULL

Create a view that displays the patient's full name, appointment date & time, doctor's full name, & department name for all scheduled appointments

MULTI-TABLE VIEW

```
CREATE VIEW Scheduled_Appointments AS
SELECT CONCAT(Patients.FirstName, ' ' ,Patients.LastName) AS PatientName,
Appointments.AppointmentDate, Appointments.AppointmentTime,
CONCAT (Doctors.FirstName, ' ' ,Doctors.LastName) AS DoctorName, Departments.DepartmentName
FROM Patients,Appointments,Doctors,Departments
WHERE Departments.DepartmentID = Doctors.DepartmentID
AND Appointments.DoctorID = Doctors.DoctorID
AND Appointments.PatientID = Patients.PatientID
AND Appointments.Status = 'scheduled';
```

USE OF CREATE , VIEW , SELECT, CONCAT, AS, WHERE, AND, AS

RESULT

PatientName	AppointmentDate	AppointmentTime	DoctorName	DepartmentName
John Smith	2024-05-12	10:00:00	Robert Johnson	Cardiology
Emily Johnson	2024-05-13	11:00:00	Emily Williams	Pediatrics
David Martinez	2024-05-16	14:00:00	David Martinez	Neurology

Views for doctor's schedule

```
CREATE VIEW DoctorSchedules AS
SELECT d.FirstName AS DoctorFirstName, d.LastName AS DoctorLastName, d.Specialty,
       p.FirstName AS PatientFirstName, p.LastName AS PatientLastName,
       a.AppointmentDate, a.AppointmentTime, a.Status
FROM Doctors d
JOIN Appointments a ON d.DoctorID = a.DoctorID
JOIN Patients p ON a.PatientID = p.PatientID;
```

USE OF CREATE , VIEW , SELECT, INNER JOIN

RESULT

DoctorFirstName	DoctorLastName	Specialty	PatientFirstName	PatientLastName	AppointmentDate	AppointmentTime	Status
Robert	Johnson	Cardiologist	John	Smith	2024-05-12	10:00:00	scheduled
Robert	Johnson	Cardiologist	John	Smith	2024-05-17	15:00:00	completed
Emily	Williams	Pediatrician	Emily	Johnson	2024-05-13	11:00:00	scheduled
Michael	Davis	Orthopedic Surgeon	Michael	Brown	2024-05-14	12:00:00	completed
Sarah	Garcia	Dermatologist	Sarah	Davis	2024-05-15	13:00:00	canceled
David	Martinez	Neurologist	David	Martinez	2024-05-16	14:00:00	scheduled

Create a trigger that automatically updates the staff count of a department in the Departments table whenever a new doctor is added or removed. Also use Union to combine two Queries for both insertion & deletion.

```
DELIMITER $$
CREATE TRIGGER UPDATE_STAFFCOUNT_AFTER_INSERT
AFTER INSERT ON DOCTORS
FOR EACH ROW
BEGIN
IF NEW.DEPARTMENTID IS NOT NULL THEN
UPDATE DEPARTMENTS
SET STAFFCOUNT = STAFFCOUNT + 1
WHERE DEPARTMENTID = NEW.DEPARTMENTID;
    END IF;
END;
$$
DELIMITER ;
UNION
DELIMITER $$
CREATE TRIGGER UPDATE_STAFFCOUNT_AFTER_DELETE
AFTER DELETE ON DOCTORS
FOR EACH ROW
BEGIN
IF OLD.DEPARTMENTID IS NOT NULL THEN
UPDATE DEPARTMENTS
SET STAFFCOUNT = STAFFCOUNT - 1
WHERE DEPARTMENTID = OLD.DEPARTMENTID;
    END IF;
END;
$$
DELIMITER ;
```

USE OF CREATE,TRIGGER, IF, THEN, UPDATE, UNION, SET

```
-- Query to check the trigger
-- display staff count first
SELECT DepartmentID, StaffCount FROM Departments WHERE DepartmentID
```

DepartmentID	StaffCount
201	15
NULL	NULL

```
-- Insert a new doctor into the department with DepartmentID 201
INSERT INTO Doctors (DepartmentID, FirstName, LastName, Specialty, ContactNumber, Email)
VALUES (201, 'Test', 'Doctor', 'Test Specialty', '123-456-7890', 'test.doctor@hc.com');

-- Check the updated staff count for the department
SELECT DepartmentID, StaffCount FROM Departments WHERE DepartmentID = 201;
-- staff count updated.
```

RESULT

DepartmentID	StaffCount
201	16
NULL	NULL

Trigger Logs every time an appointment status is updated, which is helpful for auditing & tracking patient flow

```sql
CREATE TABLE AppointmentLog (
    LogID INT AUTO_INCREMENT PRIMARY KEY,
    AppointmentID INT,
    OldStatus ENUM('scheduled', 'completed', 'canceled'),
    NewStatus ENUM('scheduled', 'completed', 'canceled'),
    UpdateTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- trigger to update appoinment log when appointment table is updated.
DELIMITER $$
CREATE TRIGGER LogAppointmentUpdate
AFTER UPDATE ON Appointments
FOR EACH ROW
BEGIN
    IF OLD.Status != NEW.Status THEN
        INSERT INTO AppointmentLog (AppointmentID, OldStatus, NewStatus)
        VALUES (OLD.AppointmentID, OLD.Status, NEW.Status);
    END IF;
END;
$$
DELIMITER ;

-- Updating an appointment's status to test the trigger
SELECT * FROM Appointments ;

UPDATE Appointments SET Status = 'completed' WHERE AppointmentID = 405;
UPDATE Appointments SET Status = 'scheduled' WHERE AppointmentID = 405;

-- Checking the log to see if the update has been recorded
SELECT * FROM AppointmentLog;
```

USE OF CREATE, TRIGGER, IF, THEN, UPDATE

RESULT

LogID	AppointmentID	OldStatus	NewStatus	UpdateTime
1	405	scheduled	completed	2024-05-15 23:27:49
2	405	completed	scheduled	2024-05-15 23:28:25
NULL	NULL	NULL	NULL	NULL

Trigger checks inventory levels after an update & logs a warning if the quantity of any item falls below a predefined threshold, aiding in inventory management.

```sql
CREATE TABLE InventoryLog (
    LogID INT AUTO_INCREMENT PRIMARY KEY,
    ItemID INT,
    Quantity INT,
    WarningMessage VARCHAR(255),
    LogTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- trigger
DELIMITER $$
CREATE TRIGGER CheckInventoryAfterUpdate
AFTER UPDATE ON Inventory
FOR EACH ROW
BEGIN
    IF NEW.Quantity < 100 THEN
        INSERT INTO InventoryLog (ItemID, Quantity, WarningMessage)
        VALUES (NEW.ItemID, NEW.Quantity, CONCAT('Warning: Low inventory for item ID ', NEW.ItemID));
    END IF;
END;
$$
DELIMITER ;

-- Updating an inventory item to reduce its quantity below the threshold to test the trigger
UPDATE Inventory SET Quantity = 95 WHERE ItemID = 602;

-- Checking the inventory log to see if the warning has been recorded
SELECT * FROM InventoryLog;
```

USE OF CREATE, TRIGGER, IF, THEN, UPDATE, < CONDITION

RESULT

LogID	ItemID	Quantity	WarningMessage	LogTime
1	602	95	Warning: Low inventory for item ID 602	2024-05-15 23:31:48
2	602	95	Warning: Low inventory for item ID 602	2024-05-15 23:51:00
NULL	NULL	NULL	NULL	NULL

Trigger Output

```
Show Triggers;
```

USE OF SHOW

RESULT

Trigger	Event	Table	Statement	Timing	Created	sql_mode	Definer	character_set_client	collation_connection	Database Collation
LogStatusChange	UPDATE	appointments	BEGIN IF NEW.Status <> OLD.Status ...	AFTER	2024-05-15 22:01:15.19	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@localhost	utf8mb4	utf8mb4_0900_ai_ci	utf8mb4_0900_ai_ci
UPDATE_STAFFCOUNT_AFTER_DELETE	DELETE	doctors	BEGIN IF OLD.DEPARTMENTID IS NOT N...	AFTER	2024-05-15 21:29:54.59	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@localhost	utf8mb4	utf8mb4_0900_ai_ci	utf8mb4_0900_ai_ci
CheckInventoryLevels	UPDATE	inventory	BEGIN IF NEW.Quantity < 50 THEN ...	AFTER	2024-05-15 22:01:18.37	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@localhost	utf8mb4	utf8mb4_0900_ai_ci	utf8mb4_0900_ai_ci

Get a list of items in the inventory along with their quantities & the department they belong to.

```
SELECT ItemName, Quantity, ExpiryDate, DepartmentName

FROM Inventory

INNER JOIN Departments ON Inventory.DepartmentID = Departments.DepartmentID;
```

USE OF INNER JOIN

RESULT

ItemName	Quantity	ExpiryDate	DepartmentName
Bandages	200	2025-12-31	Orthopedics
Aspirin	500	2023-06-30	Cardiology
Antibiotics	100	2024-09-30	Pediatrics
Anti-Allergics	50	2025-03-31	Dermatology
Apomorphine	300	2024-12-31	Neurology

Update the diagnosis & prescription for the medical record with the RecordID 503, setting the diagnosis to 'Fractured Arm (Updated)' & the prescription to 'Surgery completed, Bed rest recommended'

```
UPDATE MedicalRecords

SET

Diagnosis = 'Fractured Arm (Updated)',

Prescription = 'Surgery completed, Bed rest recommended'

WHERE RecordID = 503;
```

USE OF UPDATE, SET, WHERE

RESULT

	RecordID	PatientID	DoctorID	DateOfVisit	Diagnosis	Prescription
▶	501	101	301	2024-05-12	Hypertension	Medication A
	502	102	302	2024-05-13	Common Cold	Rest and Fluids
	503	103	303	2024-05-14	Fractured Arm (Updated)	Surgery completed, Bed rest recommended
	504	104	304	2024-05-15	Skin Allergy	Topical cream prescribed
	505	105	305	2024-05-16	Migraine	Pain relievers
*	NULL	NULL	NULL	NULL	NULL	NULL

ENHANCING HEALTHCARE THROUGH DATA MANAGEMENT

1. Enhances operational efficiencies
2. Real-time decision-making
3. Improves patient outcome
4. Database robust and reliable

Thank you !