a)

Patient Management: The system should allow for the creation and management of patient records including personal information, medical history, treatments

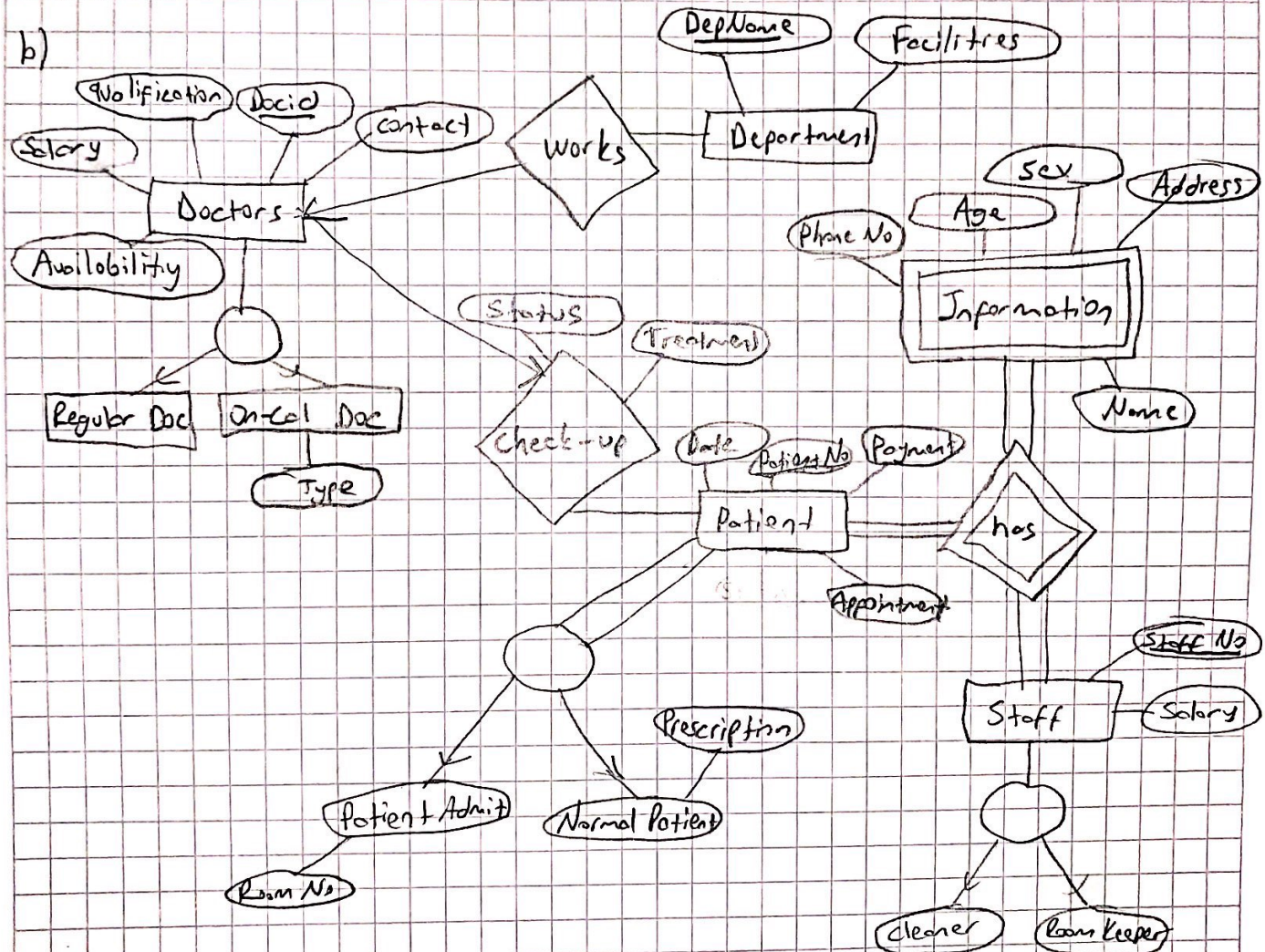
Staff Management: The system shoul enable the management of staff information, including scheduling, workload, performance and training.

Resource Management: The system should track and manage the hospital's resources, such as medical equipment, pharmaceutical supplies, and bed availability

Billing and Payment: The system should provide and integrated billing and payment system for patients, insurance companies and other stakeholders.

Security and Privacy: The system should ensure the security and privacy of patient and staff information

b)



c)

Doctor:

DocID → Qualification, Salary

Patient:

PatientNO → Payment, Prescription

Department:

DeptName → Facilities

Staff:

StaffNO → Salary

e)

a) CREATE FUNCTION GetPatientRecords()

RETURNS TABLE (

```
PatientNO INT,
Name VARCHAR(30),
Date DATE,
Payment INT
```

)

AS $$

BEGIN

```
RETURN QUERY
SELECT PatientNO, Name, Date, Payment
FROM Patient;
```

END;

b)

```
CREATE FUNCTION CalculateStaffAvgSalary()
RETURNS NUMERIC
AS $$
DECLARE
    totalSalary NUMERIC := 0;
    counter INT := 0;
    avgSalary NUMERIC
    empRow RECORD
BEGIN
    FOR empRow IN SELECT Salary FROM Staff
    LOOP
        totalSalary := totalSalary + empRow.Salary;
        counter := counter + 1;
    END LOOP;
        avgSalary := totalSalary / counter;
        RETURN avgSalary;
    END;
```

c)

```
CREATE FUNCTION UpdatePatientStatus(patientNO INT, newStatus VARCHAR(20))
RETURNS VARCHAR(100)
AS $$
DECLARE
    currentStatus VARCHAR(20);
BEGIN
    SELECT Status INTO currentStatus FROM Patient WHERE PatientNo = patientNo;
    UPDATE Patient SET Status = newStatus WHERE PatientNo = patientNO;
    RETURN 'Patient' || patientNO || ' status updates from' || currentStatus || '
    to' || newStatus;
END;
```

f)

a)

```
CREATE TRIGGER UpdatePatientStatusTrigger
AFTER UPDATE ON Patient
REFERENCING OLD ROW AS oldRow, NEW ROW AS newRow
FOR EACH ROW
    WHEN (newRow.Status = 'Discharged' AND oldRow.Status <> 'Discharged')
EXECUTE FUNCTION UpdatePatientStatistics();
```

b)

```
CREATE TRIGGER UpdatePatientPayment
AFTER UPDATE OF Payment ON Patient
FOR EACH ROW
    WHEN (NEW.Payment <> OLD.Payment)
EXECUTE FUNCTION NotifyPaymentChange();
```

c)

```
CREATE TRIGGER PatientPrescriptionTrigger
BEFORE INSERT ON Patient
FOR EACH ROW
EXECUTE FUNCTION CheckPatientStatus();
```

d)

```
DROP TRIGGER [IF EXISTS] trigger_name [ON table_name];
Example; DROP TRIGGER IF EXISTS UpdatePatientStatusTrigger ON Patient;

SHOW TRIGGERS [FROM db_name]
 [LIKE 'pattern' | WHERE expr]

SELECT *
FROM information_schema.triggers
WHERE trigger_schema = 'public';
```

f)

```
BEGIN TRANSACTION;
-- Update patient information

UPDATE Patients

SET Name = 'Alperen Karagetle', Address = 'Istanbul Tuzla, Phone = '555-1234'
WHERE patientNO = 1;
COMMIT;

BEGIN TRANSACTION;

-- Update doctor's availability
UPDATE Doctors
SET Availability = 'Busy'
WHERE DocID = 99;

--Notify patients about schedule change
 UPDATE Appointment
 SET Notification = 'Your appointment with Doctor Alperen has been rescheduled.
 WHERE DocID = 99 AND Date >= CURRENT_DATE;

  COMMIT;
```