```sql
DROP TABLE IF EXISTS Administrator;
DROP TABLE IF EXISTS Patient;
DROP TABLE IF EXISTS AdmissionType;
DROP TABLE IF EXISTS Department;
DROP TABLE IF EXISTS Admission;

CREATE TABLE Administrator (
    UserName VARCHAR(10) PRIMARY KEY,
    Password VARCHAR(20) NOT NULL,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Email VARCHAR(20) NOT NULL
);

INSERT INTO Administrator VALUES
('jdoe', 'Pass1234', 'John', 'Doe', 'jdoe@csh.com'),
('jsmith', 'Pass5678', 'Jane', 'Smith', 'jsmith@csh.com'),
('ajohnson', 'Passabcd', 'Alice', 'Johnson', 'ajohnson@csh.com'),
('bbrown', 'Passwxyz', 'Bob', 'Brown', 'bbrown@csh.com'),
('cdavis', 'Pass9876', 'Charlie', 'Davis', 'cdavis@csh.com'),
('ksmith', 'Pass5566', 'Karen', 'Smith', 'ksmith@csh.com');

CREATE TABLE Patient (
    PatientID VARCHAR(10) PRIMARY KEY,
    Password VARCHAR(20) NOT NULL,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Mobile VARCHAR(20) NOT NULL
);

INSERT INTO Patient VALUES
('dwilson', 'Pass5432', 'David', 'Wilson', '4455667788'),
('etylor', 'Passlmno', 'Eva', 'Taylor', '5566778899'),
('faderson', 'Passrstu', 'Frank', 'Anderson', '6677889900'),
('gthomas', 'Pass1357', 'Grace', 'Thomas', '7788990011'),
('smartinez', 'Pass2468', 'Stan', 'Martinez', '8899001122'),
('lroberts', 'Pass1122', 'Laura', 'Roberts', '9900112233');


CREATE TABLE AdmissionType (
    AdmissionTypeID SERIAL PRIMARY KEY,
    AdmissionTypeName VARCHAR(20) UNIQUE NOT NULL
);

INSERT INTO AdmissionType VALUES (1, 'Emergency');
INSERT INTO AdmissionType VALUES (2, 'Transfer');
INSERT INTO AdmissionType VALUES (3, 'Inpatient');
INSERT INTO AdmissionType VALUES (4, 'Outpatient');

CREATE TABLE Department (
    DeptId SERIAL PRIMARY Key,
    DeptName VARCHAR(20) UNIQUE not NULL
);

INSERT INTO Department VALUES (1, 'General');
INSERT INTO Department VALUES (2, 'Emergency');
INSERT INTO Department VALUES (3, 'Surgery');
INSERT INTO Department VALUES (4, 'Obstetrics');
INSERT INTO Department VALUES (5, 'Rehabilitation');
INSERT INTO Department VALUES (6, 'Paediatrics');

CREATE table Admission (
    AdmissionID SERIAL PRIMARY KEY,
    AdmissionType INTEGER NOT NULL,
    Department INTEGER NOT NULL,
      Patient VARCHAR(10) NOT NULL,
      Administrator VARCHAR(10) NOT NULL,
    Fee Decimal(7,2),
    DischargeDate Date,
    Condition VARCHAR(500),
      FOREIGN KEY(AdmissionType) REFERENCES AdmissionType,
      FOREIGN KEY(Department) REFERENCES Department,
      FOREIGN KEY(Patient) REFERENCES Patient,
      FOREIGN KEY(Administrator) REFERENCES Administrator
);

INSERT INTO Admission (AdmissionType, Department, Fee, Patient, Administrator,
DischargeDate, Condition) VALUES
    (4, 1, 666.00, 'lroberts', 'jdoe', '2024-02-28', 'a red patch on my skin that
looks irritated. It started small but has been spreading and feels warm to the
touch'),
      (2, 1, 100.00, 'gthomas', 'jdoe', '2021-09-11', NULL),
      (1, 2, NULL, 'lroberts','jsmith', '2019-09-02', 'Admitted to the emergency
department after suffering head trauma from a fall, requiring a CT scan and
observation for potential concussion.'),
      (2, 3, 7688.00, 'dwilson','ajohnson', '2022-12-01', NULL),
      (2, 6, 1600.00, 'faderson', 'ajohnson', '2014-09-03', 'Child admitted to the
hospital with a severe asthma attack, requiring oxygen therapy and nebulizer
treatment.'),
      (4, 1, 90.00, 'gthomas', 'ksmith', '2021-07-04', 'Routine follow-up
consultation to review progress after recent knee surgery, with positive recovery
observed.'),
      (1, 2, 1450.00, 'smartinez', 'jsmith', NULL, 'Admitted to the emergency
department with severe food poisoning, requiring IV fluids and anti-nausea
medication for recovery.'),
      (4, 5, 180.95, 'dwilson', 'cdavis', '2021-11-06', 'Attended a physiotherapy
session as part of an ongoing rehabilitation program following shoulder surgery.'),
      (3, 1, 2000.00, 'etylor', 'ajohnson', '2021-09-10', NULL),
      (2, 4, 8290.00, 'gthomas', 'jsmith', '2024-09-01', 'Postpartum care following
a natural childbirth, including monitoring of both the mother and the newborn for
potential complications.'),
      (2, 6, 1800.00, 'faderson', 'bbrown',  NULL, 'Child admitted to the
paediatrics department for severe pneumonia, requiring intravenous antibiotics and
respiratory therapy.'),
      (4, 1, 75.00, 'gthomas', 'bbrown', '2023-11-19', 'Routine general
practitioner consultation for a follow-up after a recent bout of seasonal
allergies.'),
      (3, 3, 7000.50, 'smartinez', 'jdoe', '2024-10-15', NULL),
      (1, 2, NULL, 'etylor', 'jdoe', NULL, 'I am having intense, crushing pain in
my chest that feels like an elephant is sitting on it. It is spreading to my left
arm and neck.');


--STORED FUNCTIONS :-

--1. Stored function for login
CREATE OR REPLACE FUNCTION checkLogin(
      --attributes of checkLogin
    p_username VARCHAR(10),
    p_password VARCHAR(20)
) RETURNS TABLE (
      --return values of checkLogin
    username VARCHAR(10),
    firstName VARCHAR(50),
    lastName VARCHAR(50),
    email VARCHAR(50)
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        Administrator.UserName,
        Administrator.FirstName,
        Administrator.LastName,
        Administrator.Email
    FROM
        Administrator
    WHERE
        Administrator.UserName = p_username
        AND Administrator.Password = p_password;     -- Checks whether
administrator exists in database

    -- If no rows are found, return nothing
END;
$$ LANGUAGE plpgsql;

--2. Stored function to find admissions by admin
CREATE OR REPLACE FUNCTION findAdmissionsByAdmin(
      --attributes of findAdmissionByAdmin
    p_admin VARCHAR(10)
) RETURNS TABLE (
    --return values of findAdmissionByAdmin
      admission_ID INTEGER,
    admission_type VARCHAR,
    admission_department VARCHAR,
      discharge_date VARCHAR,
      fee VARCHAR,
    patient VARCHAR,
    condition VARCHAR(500)
) AS $$
BEGIN
    RETURN QUERY
    SELECT
            a.AdmissionID,
        at.AdmissionTypeName,
        d.DeptName,
        COALESCE(TO_CHAR(a.DischargeDate, 'DD-MM-YYYY')::VARCHAR, '') AS
discharge_date,  -- Format date to DD-MM-YYYY
        COALESCE(ROUND(a.Fee::NUMERIC, 2)::VARCHAR, '') AS fee,  -- Rounding off
fee by 2
        CONCAT(p.FirstName, ' ', p.LastName)::VARCHAR AS PatientFullName, --
Concatenating firstname and lastname
        COALESCE(a.Condition, '') AS condition  -- Replace NULL with an empty
string
    FROM
        Admission a
    JOIN
        Patient p ON a.Patient = p.PatientID
    JOIN
        AdmissionType at ON a.AdmissionType = at.AdmissionTypeID
    JOIN
        Department d ON a.Department = d.DeptID
    WHERE
        a.Administrator = p_admin  -- Joining tables inorder to retreive admission
info
      ORDER BY
            a.DischargeDate IS NULL, -- Brings the admissions with no dicharge date
at bottom
            a.DischargeDate DESC, -- Making the remaining discharge date in
descending order
            PatientFullName ASC, -- Making the patient full name in ascending order
            at.AdmissionTypeName DESC; -- Making the admission type name in
descending order
END;
$$ LANGUAGE plpgsql;

--3. Stored function to find admissions by criteria
CREATE OR REPLACE FUNCTION findAdmissionsByCriteria(
      --attributes of findAdmissionsByCriteria
    p_string VARCHAR
) RETURNS TABLE (
      --return values of findAdmissionsByCriteria
      admission_ID INTEGER,
    admission_type VARCHAR,
    admission_department VARCHAR,
      discharge_date VARCHAR,
      fee VARCHAR,
    patient VARCHAR,
    condition VARCHAR(500)
) AS $$
BEGIN
    RETURN QUERY
    SELECT
            a.AdmissionID,
        at.AdmissionTypeName,
        d.DeptName,
        COALESCE(TO_CHAR(a.DischargeDate, 'DD-MM-YYYY')::VARCHAR, '') AS
discharge_date,  -- Formatting date to DD-MM-YYYY
        COALESCE(ROUND(a.Fee::NUMERIC, 2)::VARCHAR, '') AS fee,  -- Rounding off
fee by 2
        CONCAT(p.FirstName, ' ', p.LastName)::VARCHAR AS PatientFullName, --
Concatenating first and last name
        COALESCE(a.Condition, '') AS condition  -- Replace NULL with an empty
string
    FROM
        Admission a
    JOIN
        Patient p ON a.Patient = p.PatientID
    JOIN
        AdmissionType at ON a.AdmissionType = at.AdmissionTypeID
    JOIN
        Department d ON a.Department = d.DeptID-- Joining tables inorder to
retreive searched admission info
    WHERE
            (
                  LOWER(at.AdmissionTypeName) LIKE '%' || LOWER(p_string) || '%' --
Lowering the admissionTypeName and p_string to make them case insensitive
```

```
                  OR LOWER(d.DeptName) LIKE '%' || LOWER(p_string) || '%' --
Lowering the DeptName and p_string to make them case insensitive
                  OR LOWER(a.Condition) LIKE '%' || LOWER(p_string) || '%' --
Lowering the Condition and p_string to make them case insensitive
            OR LOWER(CONCAT(p.FirstName, ' ', p.LastName)) LIKE '%' ||
LOWER(p_string) || '%' -- Lowering the fullname and p_string to make them case
insensitive
                  -- String is searched in admission type name, department name,
condtion and patient name
            )
            AND (a.DischargeDate IS NULL OR a.DischargeDate >= CURRENT_DATE -
INTERVAL '2 years') -- Making the interval of 2 years
      ORDER BY
            a.DischargeDate IS NULL DESC, -- Brings the admissions with no dicharge
date at top
            a.DischargeDate ASC, -- Making the remaining discharge dates is
ascending order
            PatientFullName ASC; -- Making the patient full name in ascending order
END;
$$ LANGUAGE plpgsql;

--4. Stored function to add a new admission
CREATE OR REPLACE FUNCTION addAdmission(
      --attributes of addAdmission
    p_admission_type VARCHAR,
    p_department VARCHAR,
    p_patient VARCHAR,
    p_condition VARCHAR,
    p_admin VARCHAR
) RETURNS VOID AS $$
DECLARE
    v_admission_type_id INTEGER;
    v_department_id INTEGER;
    v_admin_id VARCHAR(10);
BEGIN
    p_admission_type := INITCAP(p_admission_type); -- Making the first letter of
each word in uppercase and rest in lowercase
    p_department := INITCAP(p_department); -- Making the first letter of each word
in uppercase and rest in lowercase
    p_patient := LOWER(p_patient); -- Making the patient username in lowercase

    -- Get the admission type ID from the name
    SELECT AdmissionTypeID INTO v_admission_type_id
    FROM AdmissionType
    WHERE LOWER(AdmissionTypeName) = LOWER(p_admission_type);

    -- Get the department ID from the name
    SELECT DeptID INTO v_department_id
    FROM Department
    WHERE LOWER(DeptName) = LOWER(p_department);

    -- Check if the patient exists
    IF NOT EXISTS (SELECT 1 FROM Patient WHERE LOWER(PatientID) = p_patient) THEN
        RAISE EXCEPTION 'Patient % not found', p_patient;
    END IF;

    -- Check if the administrator exists
    IF NOT EXISTS (SELECT 1 FROM Administrator WHERE UserName = p_admin) THEN
        RAISE EXCEPTION 'Administrator % not found', p_admin;
    END IF;

    -- Insert the new admission
    INSERT INTO Admission (AdmissionType, Department, Patient, Administrator,
Condition)
    VALUES (v_admission_type_id, v_department_id, p_patient, p_admin, p_condition);

END;
$$ LANGUAGE plpgsql;

--5. Stored function to update the admissions
CREATE OR REPLACE FUNCTION updateAdmission(
      --attributes of updateAdmission
    p_admission_id INT,
    p_admission_type VARCHAR(20),
    p_department VARCHAR(20),
    p_discharge_date DATE,
    p_fee DECIMAL(7,2),
    p_patient VARCHAR(10),
    p_condition VARCHAR(500)
)
RETURNS VOID AS $$
DECLARE
    v_admission_type_id INT;
    v_department_id INT;
BEGIN
    p_admission_type := INITCAP(p_admission_type); -- Making the first letter of
each word in uppercase and rest in lowercase
    p_department := INITCAP(p_department); -- Making the first letter of each word
in uppercase and rest in lowercase
    p_patient := LOWER(p_patient); -- Making the patient username in lowercase

    -- Get the AdmissionType ID based on the provided admission type name
    SELECT admissiontypeid INTO v_admission_type_id
    FROM AdmissionType
    WHERE LOWER(admissiontypename) = LOWER(p_admission_type);

    IF v_admission_type_id IS NULL THEN
        RAISE EXCEPTION 'Admission type % not found', p_admission_type;
    END IF;

    -- Get the Department ID based on the provided department name
    SELECT deptid INTO v_department_id
    FROM Department
    WHERE LOWER(deptname) = LOWER(p_department);

    IF v_department_id IS NULL THEN
        RAISE EXCEPTION 'Department % not found', p_department;
    END IF;

      -- Check if the patient exists
    IF NOT EXISTS (SELECT 1 FROM Patient WHERE LOWER(PatientID) = p_patient) THEN
        RAISE EXCEPTION 'Patient % not found', p_patient;
    END IF;

    -- Updating the admission record in the Admission table
    UPDATE Admission
    SET admissiontype = v_admission_type_id,
        department = v_department_id,
        dischargedate = p_discharge_date,
        fee = p_fee,
        patient = p_patient,
        condition = p_condition
    WHERE admissionid = p_admission_id;

END;
$$ LANGUAGE plpgsql;
```