**HOSPITAL SAN ROQUE**

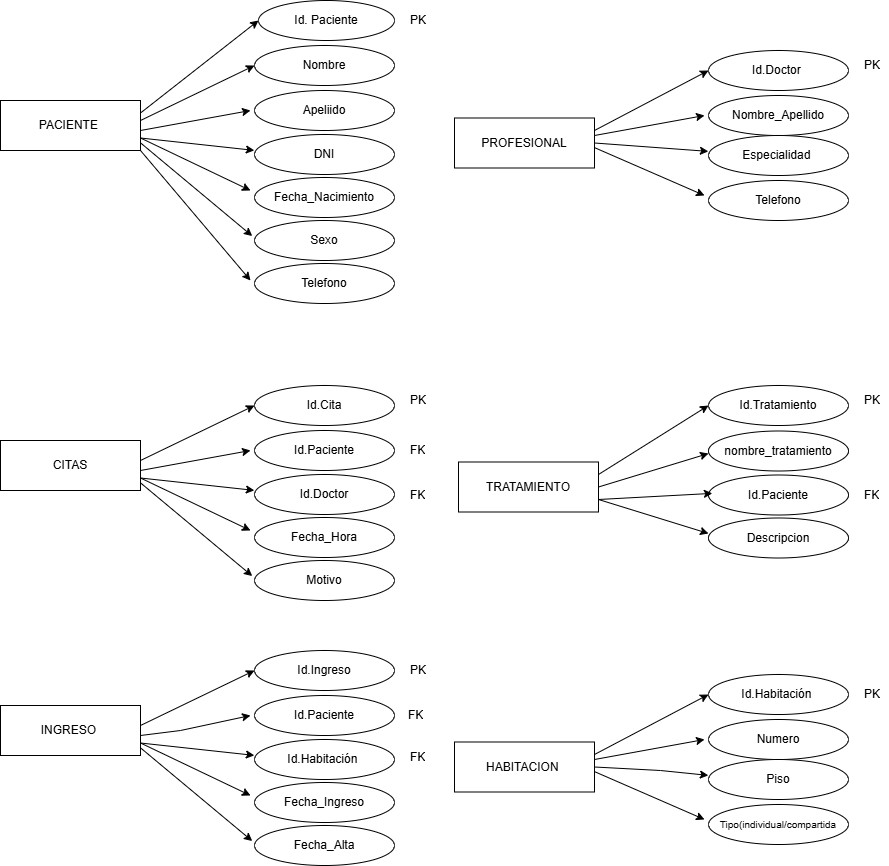
El Hospital San Roque SA. es un establecimiento medico de consultorios e internación; adquirido hace poco tiempo por una nueva firma y requiere una nueva base de datos para su mayor comodidad y orden para sus profesionales y pacientes.

La base de datos que ellos requieren, es una orden detallada día a día de todas sus consultas, internaciones y tratamientos médicos realizados.

Índice

* [Diagrama ER](#Diagrama)
* [Lista de Tablas](#tabla)
* [Triggers](#trigger)
* [Store Procedure](#store)
* [Vistas](#vista)
* [Funciones](#funcion)

Diagrama Entidad-Relación



**List****a de tablas**

**Tabla Paciente:**  En esta tabla encontraremos los datos del paciente.

Id. Paciente (PRIMARY KEY) - Nombre – Apellido – Dni - Fecha\_nacimiento – Sexo – Teléfono

**Tabla Profesional:**  En esta tabla encontraremos los datos del doctor/profesional a cargo y su especialidad.

Id. Doctor (PRIMARY KEY) – Nombre\_Apellido – Especialidad – Telefono

**Tabla Citas:** En esta tabla encontraremos los datos de la consulta del paciente; como fecha, hora y motivo de la cita.

Id. Cita (PRIMARY KEY) – Id. Paciente (FOREIGN KEY) – Id. Doctor (FOREIGN KEY) – Fecha\_hora – Motivo

**Tabla Tratamiento:** En esta tabla encontraremos la/s patologías y diagnóstico del paciente.

Id. Tratamiento (PRIMARY KEY) – Nombre tratamiento - Id. Paciente (FOREIGN KEY) – Descripción

**Tabla Ingreso:** En esta tabla encontraremos todo lo relacionado al ingreso del paciente en caso de internación, fecha y alta del mismo.

Id. Ingreso (PRIMARY KEY) – Id. Paciente (FOREIGN KEY) – Id. Habitación (FOREIGN KEY) – Fecha\_ingreso – Fecha\_alta

**Tabla Habitación:** En esta tabla encontraremos los datos de la habitación del paciente internado.

Id. Habitación (PRIMARY KEY) – Id Ingreso (FOREIGN KEY) – Piso – Número\_hab – Tipo (individual/compartida)

|  |  |  |
| --- | --- | --- |
|  | ***PACIENTE*** |  |
| **CAMPO** | **TIPO DE DATO** | **TIPO DE CLAVE** |
| ID\_PACIENTE | INT AUTO\_INCREMENT NOT NULL | PRIMARY KEY |
| NOMBRE | VARCHAR NOT NULL |  |
| APELLIDO | VARCHAR NOT NULL |  |
| DNI | INT UNIQUE NOT NULL |  |
| FECHA\_NACIMIENTO | DATE NOT NULL |  |
| SEXO | VARCHAR NOT NULL |  |
| TELEFONO | VARCHAR NOT NULL |  |

|  |  |  |
| --- | --- | --- |
|  | ***PROFESIONAL*** |  |
| **CAMPO** | **TIPO DE DATO** | **TIPO DE CLAVE** |
| ID\_DOCTOR | INT AUTO\_INCREMENT NOT NULL | PRIMARY KEY |
| NOMBRE\_APELLIDO | VARCHAR NOT NULL |  |
| ESPECIALIDAD | VARCHAR NOT NULL |  |
| TELEFONO | VARCHAR NOT NULL |  |

|  |  |  |
| --- | --- | --- |
|  | ***CITAS*** |  |
| **CAMPO** | **TIPO DE DATO** | **TIPO DE CLAVE** |
| ID\_CITA | INT AUTO\_INCREMENT NOT NULL | PRIMARY KEY |
| ID\_PACIENTE | INT NOT NULL | FOREIGN KEY |
| ID\_DOCTOR | INT NOT NULL | FOREIGN KEY |
| FECHA\_HORA | DATETIME DEFAULT CURRENT\_TIMESTAMP |  |
| MOTIVO | VARCHAR NOT NULL |  |

|  |  |  |
| --- | --- | --- |
|  | ***TRATAMIENTO*** |  |
| **CAMPO** | **TIPO DE DATO** | **TIPO DE CLAVE** |
| ID\_TRATAMIENTO | INT NOT NULL | PRIMARY KEY |
| NOMBRE\_TRATAMIENTO | VARCHAR NOT NULL |  |
| ID\_PACIENTE | INT NOT NULL | FOREIGN KEY |
| DESCRIPCION | VARCHAR NOT NULL |  |

|  |  |  |
| --- | --- | --- |
|  | ***INGRESO*** |  |
| **CAMPO** | **TIPO DE DATO** | **TIPO DE CLAVE** |
| ID\_INGRESO | INT NOT NULL | PRIMARY KEY |
| ID\_PACIENTE | INT NOT NULL | FOREIGN KEY |
| ID\_HABITACION | INT NOT NULL |  |
| FECHA\_INGRESO | DATETIME NOT NULL |  |
| FECHA\_ALTA | DATETIME NOT NULL |  |

|  |  |  |
| --- | --- | --- |
|  | ***HABITACION*** |  |
| **CAMPO** | **TIPO DE DATO** | **TIPO DE CLAVE** |
| ID\_HABITACION | INT NOT NULL | PRIMARY KEY |
| ID\_INGRESO | INT NOT NULL | FOREIGN KEY |
| PISO | VARCHAR NOT NULL |  |
| NUMERO\_HAB | INT NOT NULL |  |
| TIPO (individual/compatida) | VARCHAR NOT NULL |  |

TRIGGERS

Trigger creado para evitar que un paciente tenga dos ingresos a la vez.

DELIMITER //

CREATE TRIGGER evitar\_ingreso\_duplicado

BEFORE INSERT ON ingreso

FOR EACH ROW

BEGIN

IF EXISTS (

SELECT 1 FROM ingreso

WHERE id\_paciente = NEW.id\_paciente AND fecha\_alta IS NULL

) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'El paciente ya está ingresado.';

END IF;

END //

DELIMITER ;

Trigger creado para evitar agendar dos veces la misma cita.

DELIMITER //

CREATE TRIGGER evitar\_cita\_duplicada

BEFORE INSERT ON citas

FOR EACH ROW

BEGIN

IF EXISTS (

SELECT 1 FROM citas

WHERE id\_paciente = NEW.id\_paciente

AND id\_doctor = NEW.id\_doctor

AND DATE(fecha\_hora) = DATE(NEW.fecha\_hora)

) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'El paciente ya tiene una cita con este médico ese día.';

END IF;

END //

DELIMITER ;

Trigger creado para controlar que no se repitan tratamientos innecesariamente.

DELIMITER //

CREATE TRIGGER evitar\_tratamiento\_duplicado

BEFORE INSERT ON tratamiento

FOR EACH ROW

BEGIN

IF EXISTS (

SELECT 1 FROM tratamiento

WHERE id\_paciente = NEW.id\_paciente

AND nombre\_tratamiento = NEW.nombre\_tratamiento

) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'El paciente ya tiene este tratamiento asignado.';

END IF;

END //

DELIMITER ;

STORE PROCEDURE

Store procedure para agendar una nueva cita.

DELIMITER //

CREATE PROCEDURE Registrar\_Cita (

IN p\_id\_paciente INT,

IN p\_id\_doctor INT,

IN p\_fecha DATETIME,

IN p\_motivo VARCHAR(30)

)

BEGIN

INSERT INTO CITAS (id\_paciente, id\_doctor, fecha\_hora, motivo)

VALUES (p\_id\_paciente, p\_id\_doctor, p\_fecha, p\_motivo);

END //

DELIMITER ;

Store procedure creado para dar de alta a un paciente.

DELIMITER //

CREATE PROCEDURE Dar\_Alta\_Paciente (

IN p\_id\_ingreso INT,

IN p\_fecha\_alta DATETIME

)

BEGIN

UPDATE ingreso

SET fecha\_alta = p\_fecha\_alta

WHERE id\_ingreso = p\_id\_ingreso;

END //

DELIMITER ;

Store procedure creado para ingresar paciente a habitación.

DELIMITER //

CREATE PROCEDURE Ingresar\_Paciente (

IN p\_id\_paciente INT,

IN p\_id\_habitacion INT,

IN p\_fecha\_ingreso DATETIME

)

BEGIN

INSERT INTO ingreso (id\_paciente, id\_habitacion, fecha\_ingreso, fecha\_alta)

VALUES (p\_id\_paciente, p\_id\_habitacion, p\_fecha\_ingreso, NULL);

END //

DELIMITER ;

Store procedure creado para ingresar pacientes nuevos al hospital

DELIMITER //

CREATE PROCEDURE ingresar\_nuevo\_paciente (

IN p\_nombre VARCHAR(20),

IN p\_apellido VARCHAR(20),

IN p\_dni INT,

IN p\_fecha\_nacimiento DATE,

IN p\_sexo VARCHAR(2),

IN p\_telefono VARCHAR(15)

)

BEGIN

-- Verificar que el DNI no esté repetido

IF EXISTS (SELECT 1 FROM paciente WHERE dni = p\_dni) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Ya existe un paciente con ese DNI.';

ELSE

INSERT INTO paciente (

nombre, apellido, dni, fecha\_nacimiento, sexo, telefono

) VALUES (

p\_nombre, p\_apellido, p\_dni, p\_fecha\_nacimiento, p\_sexo, p\_telefono

);

END IF;

END //

DELIMITER ;

CALL ingresar\_nuevo\_paciente('Lucia', 'Fernandez', 40875123, '1994-07-21', 'F', '3514448899');

CALL ingresar\_nuevo\_paciente('Marcos', 'Torres', 37249865, '1987-02-10', 'M', '3516677885');

CALL ingresar\_nuevo\_paciente('Sofia', 'Herrera', 42315678, '2001-11-05', 'F', '3515511997');

CALL ingresar\_nuevo\_paciente('Pedro', 'Salinas', 38977452, '1975-03-16', 'M', '3518844556');

VISTAS

Vista creada para ver fácilmente todas las citas con nombres y especialidades.

CREATE VIEW vista\_historial\_citas AS

SELECT

c.id\_cita,

c.fecha\_hora,

c.motivo,

p.nombre AS nombre\_paciente,

p.apellido AS apellido\_paciente,

d.nombre\_apellido AS nombre\_doctor,

d.especialidad

FROM citas c

JOIN paciente p ON c.id\_paciente = p.id\_paciente

JOIN profesional d ON c.id\_doctor = d.id\_doctor;

Vista creada para ver los tratamientos que tiene cada paciente.

CREATE VIEW vista\_tratamientos\_paciente AS

SELECT

t.id\_tratamiento,

t.nombre\_tratamiento,

t.descripcion,

p.nombre,

p.apellido

FROM tratamiento t

JOIN paciente p ON t.id\_paciente = p.id\_paciente;

Vista creada para ver si la habitación esta ocupada o disponible.

CREATE VIEW vista\_ocupacion\_habitaciones AS

SELECT

h.id\_habitacion,

h.numero\_hab,

h.piso,

h.tipo,

CASE

WHEN i.fecha\_alta IS NULL THEN 'Ocupada'

ELSE 'Libre'

END AS estado

FROM habitacion h

LEFT JOIN ingreso i ON h.id\_habitacion = i.id\_habitacion

AND i.fecha\_alta IS NULL;

FUNCIONES

Función creada para obtener el nombre completo del paciente por id.

DELIMITER //

CREATE FUNCTION nombre\_completo\_paciente(p\_id INT)

RETURNS VARCHAR(50)

DETERMINISTIC

BEGIN

DECLARE nombre\_completo VARCHAR(50);

SELECT CONCAT(nombre, ' ', apellido)

INTO nombre\_completo

FROM paciente

WHERE id\_paciente = p\_id;

RETURN nombre\_completo;

END //

DELIMITER ;

Función creada para ver la cantidad de citas de un paciente.

DELIMITER //

CREATE FUNCTION total\_citas\_paciente(p\_id INT)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE total INT;

SELECT COUNT(\*) INTO total FROM CITAS WHERE id\_paciente = p\_id;

RETURN total;

END //