Instituto Politécnico Nacional

Escuela Superior de Cómputo

Unidad de aprendizaje:
Bases de Datos

Proyecto

Clínica Odontológica

Grupo: 3CV1

Integrantes:

Aguilar Ibarra José Moisés
Boleta: 2022630329

González Cárdenas Ángel Aquilez
Boleta: 2016630152

Pérez Olivares José Julio
Boleta: 2022630070

Profesor: Blanco Almazán Iván Eduardo

Índice

1. Introducción

El presente escrito documenta el proceso de resolución de una problemática particular presentada como parte del proyecto de evaluación correspondiente al semestre *2023-2* de la unidad de aprendizaje *Bases de Datos* para la carrera de *Ingeniería en Sistemas Computacionales* del programa académico *ISC2020*.

2. Problemática a resolver

Clínica odontológica

Una nueva clínica abre sus instalaciones con la problemática de que necesita un sistema que le ayude a administrar desde sus tratamientos hasta los diversos pacientes que atiende. Para ello la clínica busca almacenar toda la información referente a los diversos pacientes que atenderá. Para ello el sistema guardará cada paciente con su nombre compuesto de nombre, apellido paterno y materno, la dirección dada por calle, número, colonia y el código postal, su fecha de nacimiento por la cual el sistema deberá calcular la edad.

Este paciente podrá solicitar vía telefónica una consulta por lo que se le proporcionará la fecha de consulta y el número de su consulta. Cabe aclarar que este número de consulta ayudará a saber cuál fue la consulta en el tiempo de vida de la clínica y no en el día. Cada consulta es brindada por un odontólogo, el cual el administrador registrará su nombre completo dividido en su nombre y apellidos, así como la especialidad en el área que tiene, es por ello por lo que dentro de la clínica se le dará automáticamente un número de trabajador.

Durante cada consulta en la cual se encuentre el odontólogo, se realizarán diversos tratamientos los cuales tienen un precio. Estos tratamientos recibirán un nombre clave, ya que son distintos para cada consulta por lo que el sistema deberá de almacenar los tratamientos mediante una fecha de inicio del tratamiento y el odontólogo podrá seleccionar la opción de terminado en caso de que el tratamiento haya finalizado, lo que arrojará la fecha de fin.

Debido a que esta clínica ya tenía una sucursal cercana y se planea abrir una tercera clínica, el sistema guardará el local desde donde el odontólogo trabaja apoyándose que cada local le demos un nombre característico, así como la calle, el número y la ciudad. Por parte del odontólogo, el sistema deberá registrar el día en que fue a laborar en ese local, el horario de inicio y de fin en el que se retiró del local. Esto también aplica para cuando tenga que asistir a dicho local por caso de emergencia.

Por último, debido a que cada clínica estará conectada al sistema del local en donde se encuentre, se debe tener un registro que pueda existir más de un consultorio en cada local por lo que necesitaremos saber en todo momento con cuantos consultorios cuenta esa clínica, así como el equipo que posee. Durante la realización de cada inventario, cada equipo se registra por su número de serie, el tipo de equipo y la última fecha en la que se le dio mantenimiento.

3. Diagramas *Entidad-Relación*

3.1. *Entidad-Relación*

Después de analizar y realizar la abstracción de la problemática, se identificaron las siguientes *entidades* y sus *atributos*:

1. **Paciente**:
   a) *id_paciente*
   b) nombre_completo: nombres, apellido_paterno, apellido_materno
   c) dirección_paciente: calle, número, colonia, código_postal
   d) fecha_nacimiento, edad
   e) teléfono
2. **Consulta**:
   a) *id_consulta*
   b) fecha_consulta
3. **Odontólogo**:
   a) *id_empleado*
   b) nombre_completo: nombres, apellido_paterno, apellido_materno
   c) especialidad
4. **Tratamiento**:
   a) *id_tratamiento*
   b) nombre_clave
   c) precio
5. **Sucursal**:
   a) *id_sucursal*
   b) nombre_sucursal
   c) direccion_sucursal: calle_sucursal, numero_sucursal, ciudad_sucursal
   d) capacidad
6. **Consultorio**: *id_consultorio*
7. **Equipo**:
   a) *numero_serie*
   b) descripcion
   c) fecha_mantenimiento

Y para las *relaciones* se identificaron las siguientes:

1. Paciente - Consulta (1:N)
2. Consulta - Odontólogo (N:1)
3. Consulta - Tratamiento (1:N)
4. Odontólogo - Sucursal (N:M)
5. Sucursal - Consultorio (1:N)
6. Consultorio - Equipo (1:N)

De modo que gráficamente, se visualiza de la siguiente forma:

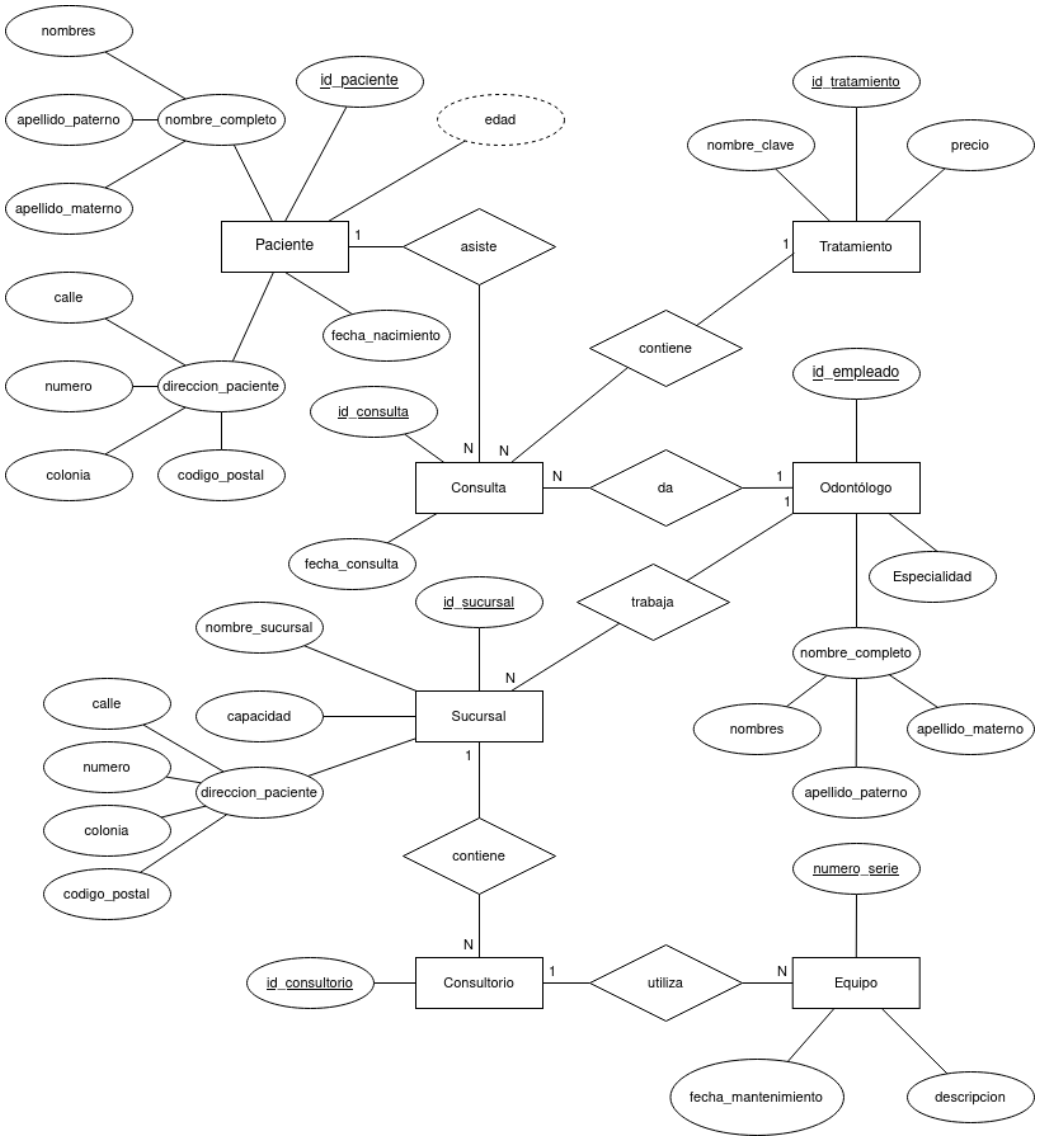

Figura 1: Diagrama Entidad-Relación

3.2. *Entidad-Relación Extendido*

Para este apartado, extendemos el caso que la problemática plantea como *emergencia*, considerándose como un tipo de *consulta*.

Por otra parte, se agrega la entidad *cita* para denotar a los contactos telefónicos que realizan los pacientes previos a la consulta.

Así, se visualiza de la siguiente forma:

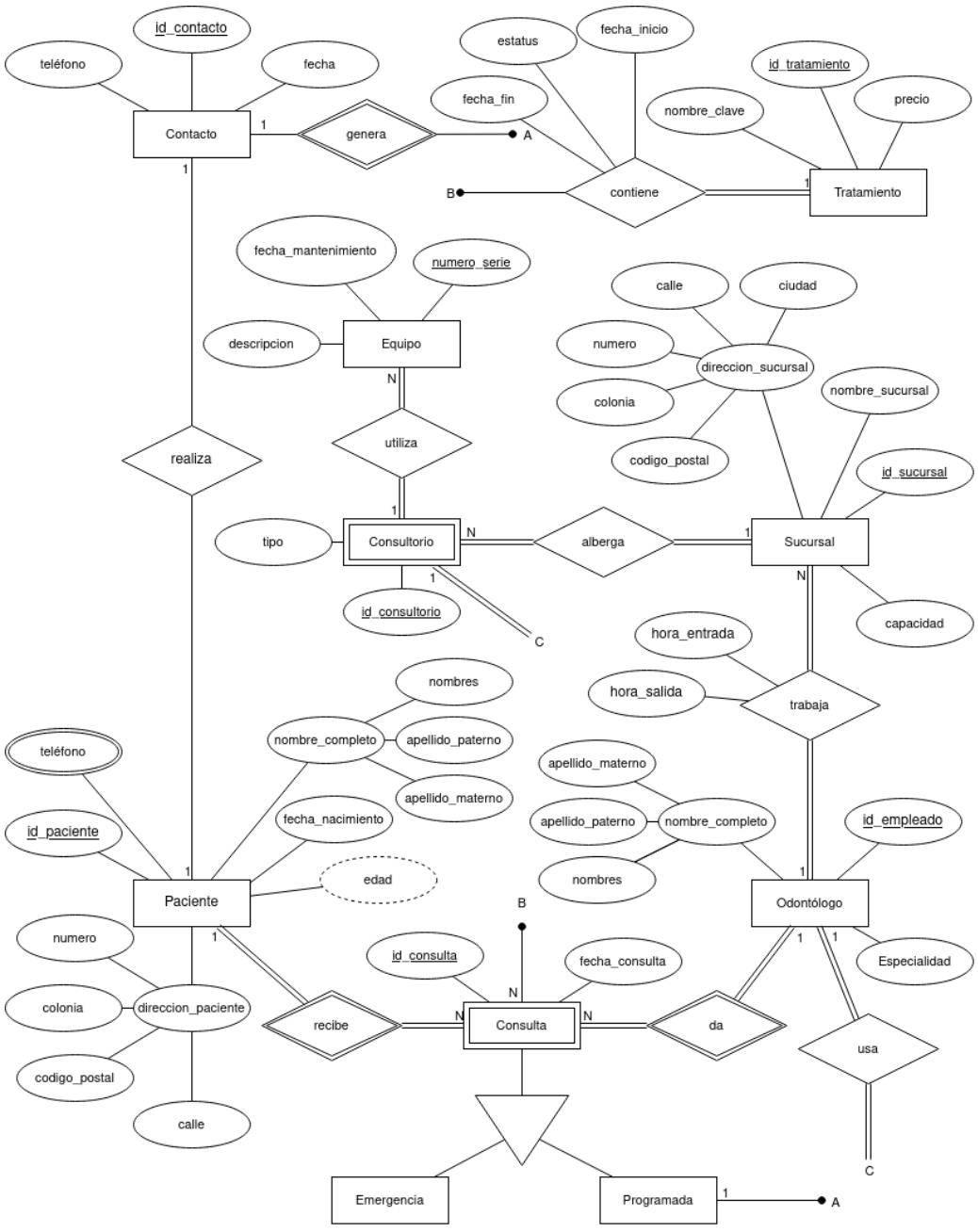

Figura 2: Diagrama Entidad-Relación

4. Diagrama Relacional

Del diagrama anterior se generó el diagrama relacional:

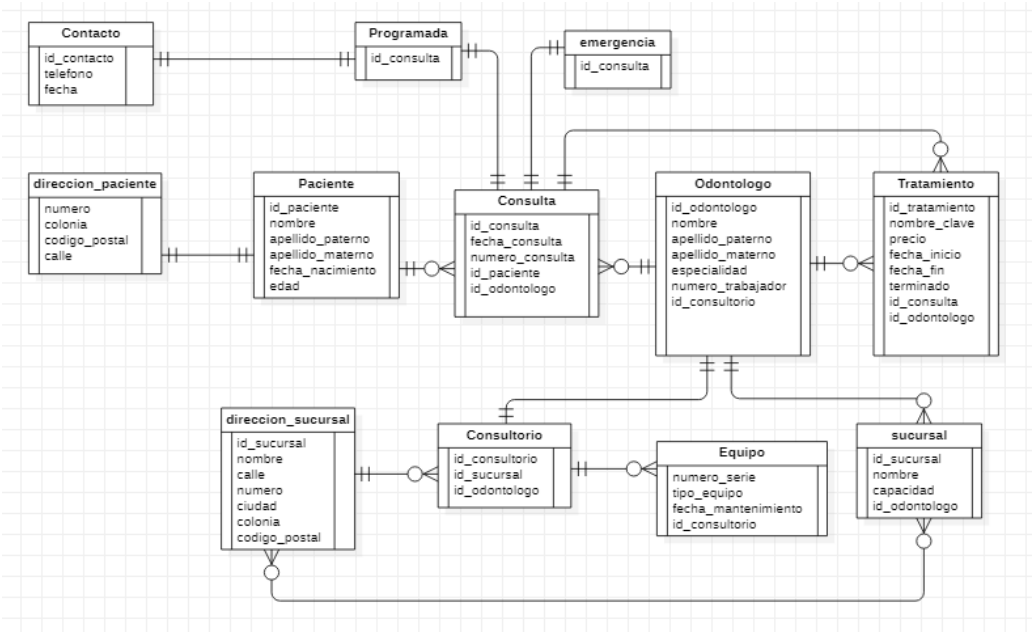

Figura 3: Diagrama Entidad-Relación

5. Diccionario de datos

De los anteriores diagramas se utilizaron los siguientes tipos de datos para su implementación en el sistema gestor de base de datos que alberga la base de datos:

Paciente
Objetivo: La tabla *paciente* alberga información relevante sobre los pacientes que acuden a la clínica odontológica.

1. id_paciente **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `nombre_paciente` texto, *no nulo*
3. `apellido_paterno` texto, *no nulo*
4. `apellido_materno` texto, *no nulo*
5. `fecha_nacimiento` fecha, *no nulo*
6. `edad` entero, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Paciente (
  id_paciente INT PRIMARY KEY AUTO_INCREMENT,
  nombre_paciente VARCHAR(50) NOT NULL,
  apellido_paterno VARCHAR(50) NOT NULL,
  apellido_materno VARCHAR(50) NOT NULL,
  fecha_nacimiento DATE NOT NULL,
  edad INT NOT NULL
);
```

Stored procedures:

```
REATE PROCEDURE sp_InsertarPaciente(
    IN nombre VARCHAR(50),
    IN apellido_paterno VARCHAR(50),
    IN apellido_materno VARCHAR(50),
    IN fecha_nacimiento DATE,
    IN edad INT,
    IN calle VARCHAR(100),
    IN numero INT,
    IN colonia VARCHAR(100),
    IN codigo_postal INT
)
BEGIN
    INSERT INTO Paciente (nombre_paciente, apellido_paterno, apellido_materno, fecha_n
    VALUES (nombre, apellido_paterno, apellido_materno, fecha_nacimiento, edad);

    SET @id_paciente = LAST_INSERT_ID();

    INSERT INTO direc_paciente (calle, numero, colonia, codigo_postal, id_paciente)
    VALUES (calle, numero, colonia, codigo_postal, @id_paciente);
END
```

direc_paciente

Objetivo: Alberga las direcciones de los pacientes y se relaciona con el paciente asociado mediante una llave foránea.

1. id_paciente **Llave foránea**
2. `calle` texto, *no nulo*
3. `numero` entero, *no nulo*
4. `colonia` texto, *no nulo*
5. `codigo_postal` fecha, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists direc_paciente (
  calle VARCHAR(100) NOT NULL,
  numero INT NOT NULL,
  colonia VARCHAR(100) NOT NULL,
  codigo_postal INT NOT NULL,
  id_paciente INT NOT NULL,
  FOREIGN KEY (id_paciente) REFERENCES Paciente(id_paciente)
);
```

Triggers:

```
CREATE TRIGGER tr_direc_paciente_delete
BEFORE DELETE ON direc_paciente
FOR EACH ROW
BEGIN
    DELETE FROM direc_paciente WHERE id_paciente = OLD.id_paciente;
END
```

Odontologo

Objetivo: Alberga información de control sobre los odontólogos.

1. numero_empleado **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `nombre` texto, *no nulo*
3. `apellido_paterno` texto, *no nulo*
4. `apellido_materno` texto, *no nulo*
5. `especialidad` texto, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Odontologo (
  id_odontologo INT PRIMARY KEY AUTO_INCREMENT,
  nombre VARCHAR(50) NOT NULL,
  apellido_paterno VARCHAR(50) NOT NULL,
  apellido_materno VARCHAR(50) NOT NULL,
```

```
  especialidad VARCHAR(100) NOT NULL,
  numero_trabajador INT NOT NULL,
  id_consultorio INT NOT NULL
);

ALTER TABLE Odontologo ADD FOREIGN KEY (id_consultorio) REFERENCES Consultorio(id_cons
```

Triggers:

```
CREATE TRIGGER tr_Odontologo_delete
BEFORE DELETE ON Odontologo
FOR EACH ROW
BEGIN
    DELETE FROM Odontologo WHERE id_consultorio = OLD.id_consultorio;
END
```

Stored procedures: Actualiza la especialidad de un odontólogo.

```
CREATE PROCEDURE sp_ActualizarEspecialidad(
    IN odontologo_id INT,
    IN nueva_especialidad VARCHAR(100)
)
BEGIN
    UPDATE Odontologo
    SET especialidad = nueva_especialidad
    WHERE id_odontologo = odontologo_id;
END
```

Consulta

Objetivo: Conjunta la información sobre los pacientes que reciben algún tratamiento, son atendidos por algún odontólogo en una hora y lugar establecidos.

1. id_consulta **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `fecha_consulta` fecha, *no nulo*
3. `id_paciente` **Llave foránea**
4. `numero_empleado` **Llave foránea**

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Consulta (
  id_consulta INT PRIMARY KEY AUTO_INCREMENT,
  fecha_consulta DATE NOT NULL,
  numero_consulta INT NOT NULL,
  id_paciente INT NOT NULL,
  id_odontologo INT NOT NULL,
  FOREIGN KEY (id_paciente) REFERENCES Paciente(id_paciente),
  FOREIGN KEY (id_odontologo) REFERENCES Odontologo(id_odontologo)
);
```

Triggers:

```
CREATE TRIGGER tr_Consulta_delete
BEFORE DELETE ON Consulta
FOR EACH ROW
BEGIN
    DELETE FROM Consulta WHERE id_paciente = OLD.id_paciente;
    DELETE FROM Consulta WHERE id_odontologo = OLD.id_odontologo;
END
```

Stored procedure: Obtiene la lista de consultas de un paciente

```
CREATE PROCEDURE sp_ConsultasPorPaciente(
    IN paciente_id INT
)
BEGIN
    SELECT *
    FROM Consulta
    WHERE id_paciente = paciente_id;
END
```

Tratamiento

Objetivo: Contiene información sobre los tratamientos, costos, duración y estatus del mismo.

1. id_tratamiento **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `nombre_clave` texto, *no nulo*
3. `precio` flotante, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Tratamiento (
  id_tratamiento INT PRIMARY KEY AUTO_INCREMENT,
  nombre_clave VARCHAR(100) NOT NULL,
  precio DECIMAL(10,2) NOT NULL,
  fecha_inicio DATE NOT NULL,
  fecha_fin DATE,
  terminado BOOLEAN NOT NULL,
  id_consulta INT NOT NULL,
  id_odontologo INT NOT NULL,
  FOREIGN KEY (id_consulta) REFERENCES Consulta(id_consulta),
  FOREIGN KEY (id_odontologo) REFERENCES Odontologo(id_odontologo)
);
```

Triggers:

```
CREATE TRIGGER tr_Tratamiento_delete
BEFORE DELETE ON Tratamiento
FOR EACH ROW
```

```
BEGIN
    DELETE FROM Tratamiento WHERE id_consulta = OLD.id_consulta;
    DELETE FROM Tratamiento WHERE id_odontologo = OLD.id_odontologo;
END
```

Sucursal

Objetivo: Conjuntar la información de las diferentes sucursales de la clínica, que contiene consultorios y equipo de forma indirecta. Capacidad hace referencia al número de consultorios que puede albergar.

1. id_sucursal **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `nombre_sucursal` texto, *no nulo*
3. `capacidad` entero, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists sucursal (
  id_sucursal INT PRIMARY KEY AUTO_INCREMENT,
  nombre_sucursal VARCHAR(100) NOT NULL,
  capacidad_sucursal INT NOT NULL,
  id_odontologo INT NOT NULL,
  FOREIGN KEY (id_odontologo) REFERENCES Odontologo(id_odontologo)
);
```

Triggers:

```
CREATE TRIGGER tr_sucursal_delete
BEFORE DELETE ON sucursal
FOR EACH ROW
BEGIN
    DELETE FROM sucursal WHERE id_odontologo = OLD.id_odontologo;
END
```

direc_sucursal

Objetivo: Alberga las direcciones de las sucursales.

1. id_sucursal **Llave foránea**
2. `calle` texto, *no nulo*
3. `numero` entero, *no nulo*
4. `colonia` texto, *no nulo*
5. `ciudad` texto, *no nulo*
6. `codigo_postal` fecha, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists direc_sucursal (
  id_sucursal INT NOT NULL,
  calle_sucursal VARCHAR(100) NOT NULL,
  numero_sucursal INT NOT NULL,
  colonia_sucursal VARCHAR(100) NOT NULL,
  ciudad_sucursal VARCHAR(100) NOT NULL,
  codigoPostal_sucursal INT NOT NULL,
  FOREIGN KEY (id_sucursal) REFERENCES sucursal(id_sucursal)
);
```

Triggers:

```
CREATE TRIGGER tr_direc_sucursal_delete
BEFORE DELETE ON direc_sucursal
FOR EACH ROW
BEGIN
    DELETE FROM direc_sucursal WHERE id_sucursal = OLD.id_sucursal;
END
```

Consultorio

Objetivo: Alberga información sobre los diferentes consultorios que se utilizan en la clínica así como el odontólogo asignado.

1. id_consultorio **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. id_sucursal **Llave foránea**
3. numero_empleado **Llave foránea**

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Consultorio (
  id_consultorio INT PRIMARY KEY AUTO_INCREMENT,
  id_sucursal INT NOT NULL,
  id_odontologo INT NOT NULL,
  FOREIGN KEY (id_sucursal) REFERENCES sucursal(id_sucursal),
  FOREIGN KEY (id_odontologo) REFERENCES Odontologo(id_odontologo)
);
```

Triggers:

```
CREATE TRIGGER tr_Consultorio_delete
BEFORE DELETE ON Consultorio
FOR EACH ROW
BEGIN
    DELETE FROM Consultorio WHERE id_sucursal = OLD.id_sucursal;
    DELETE FROM Consultorio WHERE id_odontologo = OLD.id_odontologo;
END
```

Equipo

Objetivo: Alberga información sobre el equipo que se utiliza en los consultorios.

1. numero_serie **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `tipo_equipo` texto, *no nulo*
3. `fecha_mantenimiento` fecha, *no nulo*
4. id_consultorio **Llave foránea**

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Equipo (
  numero_serie VARCHAR(50) PRIMARY KEY NOT NULL,
  tipo_equipo VARCHAR(100) NOT NULL,
  fecha_mantenimiento DATE NOT NULL,
  id_consultorio INT NOT NULL,
  FOREIGN KEY (id_consultorio) REFERENCES Consultorio(id_consultorio)
);
```

Triggers:

```
CREATE TRIGGER tr_Equipo_delete
BEFORE DELETE ON Equipo
FOR EACH ROW
BEGIN
    DELETE FROM Equipo WHERE id_consultorio = OLD.id_consultorio;
END
```

Contacto

Objetivo: Debido a que los clientes pueden llamar para agendar una cita, existe esta tabla para albergar información sobre los contactos de los pacientes que terminen en una cita programada.

1. id_contacto **Llave primaria**, entero, *no nulo*, *auto-incrementable*
2. `fecha_contacto` fecha, *no nulo*

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Contacto (
id_contacto INT NOT NULL PRIMARY KEY AUTO_INCREMENT
);
```

Cita programada

Objetivo: Una cita programada tiene identificación del paciente y una fecha a futuro. Enlaza el contacto del que se generó y la cita en la que terminó.

1. id_contacto **Llave foránea**
2. id_consulta **Llave foránea**

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Programada (
id_consulta INT NOT NULL,
id_contacto INT NOT NULL,
FOREIGN KEY (id_consulta) REFERENCES Consulta(id_consulta),
FOREIGN KEY (id_contacto) REFERENCES Contacto(id_contacto)
);
```

Triggers:

```
CREATE TRIGGER tr_Programada_delete
BEFORE DELETE ON Programada
FOR EACH ROW
BEGIN
    DELETE FROM Programada WHERE id_consulta = OLD.id_consulta;
    DELETE FROM Programada WHERE id_contacto = OLD.id_contacto;
END
```

Emergencia
Objetivo: Similar al anterior, pero con el carácter de emergencia.

1. id_contacto **Llave foránea**
2. id_consulta **Llave foránea**

Creación de la tabla con SQL:

```
CREATE TABLE if not exists Emergencia (
id_consulta INT NOT NULL,
id_contacto INT NOT NULL,
FOREIGN KEY (id_consulta) REFERENCES Consulta(id_consulta),
FOREIGN KEY (id_contacto) REFERENCES Contacto(id_contacto)
);
```

Triggers:

```
CREATE TRIGGER tr_Emergencia_delete
BEFORE DELETE ON Emergencia
FOR EACH ROW
BEGIN
    DELETE FROM Emergencia WHERE id_consulta = OLD.id_consulta;
    DELETE FROM Emergencia WHERE id_contacto = OLD.id_contacto;
END
```

6. Anexos

6.1. Creación de la base de datos

Se anexa el script mediante el cual se generó la base de datos, sus respectivas inserciones y los diferentes *triggers* y *stored procedures* que la acompañan.

6.2. Consultas SQL

Los siguientes enunciados describen ejercicios prácticos que se anexan como parte de la evaluación al proyecto:

1. Genere una vista donde visualice que odontólogo atiende a que paciente.
2. Genere un reporte donde se recupere que tratamiento realiza cada odontólogo.
3. En que clínica trabaja que odontólogos
4. Genere una función que a partir de introducir el nombre completo del paciente genere lo siguiente: Nombre_completo_Paciente, Edad, Tratamiento, Precio Nombre_completo, Odontologo, Cedula Clinica
5. Ingrese la siguiente información: Juan Ramirez Iñiguez de 38 años, ingreso a la clínica dentalink para un tratamiento de carillas, el cual tiene un costo de $500.00, para ello el odontólogo que lo atiende se llama Maria Angeles Peña con especialidad en ortodoncia.
6. Genere una función que me regrese los tratamientos que ha realizado un odontólogo a partir de su nombre y de la fecha de inicio.
7. Ingrese 10 nuevos odontólogos con información en todos los campos
8. Ingrese 10 nuevos clientes con sus tratamientos y asígnelos a los nuevos odontólogos que ingreso. Mínimo debe estar asignado a 4 de los nuevos odontólogos.