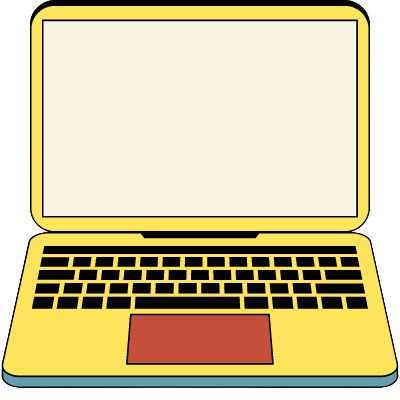
Grupo 1

Proyecto Final: Consultorio Odontológico

Curso: 5° 1ra C



Ciclo Lectivo: 2022

**Carpeta de Campo de Nicolás León Nieto Ferrari**

**Integrantes del grupo:**

* Nicolás Nieto
* Dylan Seltzer
* Kiara Chunga
* Wilver Guzmán

**Profesores:**

* Cristina Martelli -> Programación Orientada a Objetos
* Juan María McLoughlin -> Base de Datos
* Natalia González -> Análisis de Sistemas

**Nuestro Proyecto:**

Consultorio Odontológico

**Descripción breve del Proyecto:**

Se trata de una página web donde el usuario ingresa sus datos, se guardan en una base de datos, y elige si desea realizar una consulta con un odontólogo o solicitar un tratamiento.

Tras haber seleccionado alguna opción, se muestran en una tabla los turnos libres, con fecha y hora, así como el nombre del doctor.

El paciente puede seleccionar el turno que desea y se guardan los datos modificados del turno en la base de datos.

**Bitácora de Trabajo:**

21/10/22 (Grupal) **GITHUB**:

El primer paso del proyecto fue la creación del repositorio público en GitHub, así como las ramas con los nombres de cada uno de los participantes.

Se puede acceder desde la página web:

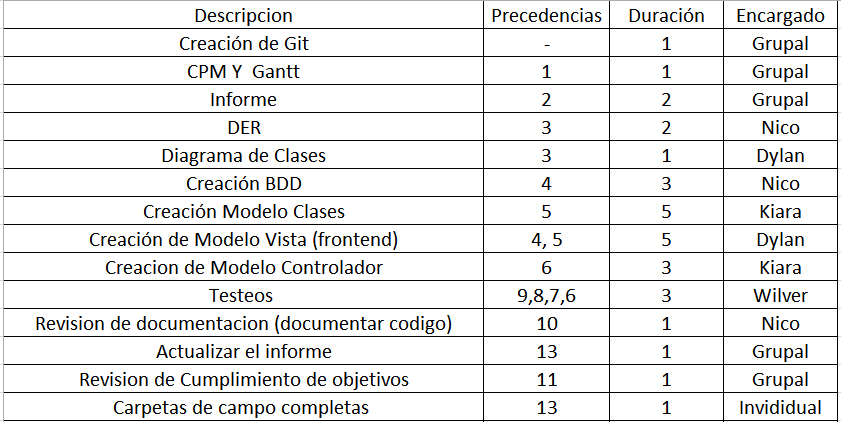
<https://github.com/nietonicolasleon/Consultorio_G1>

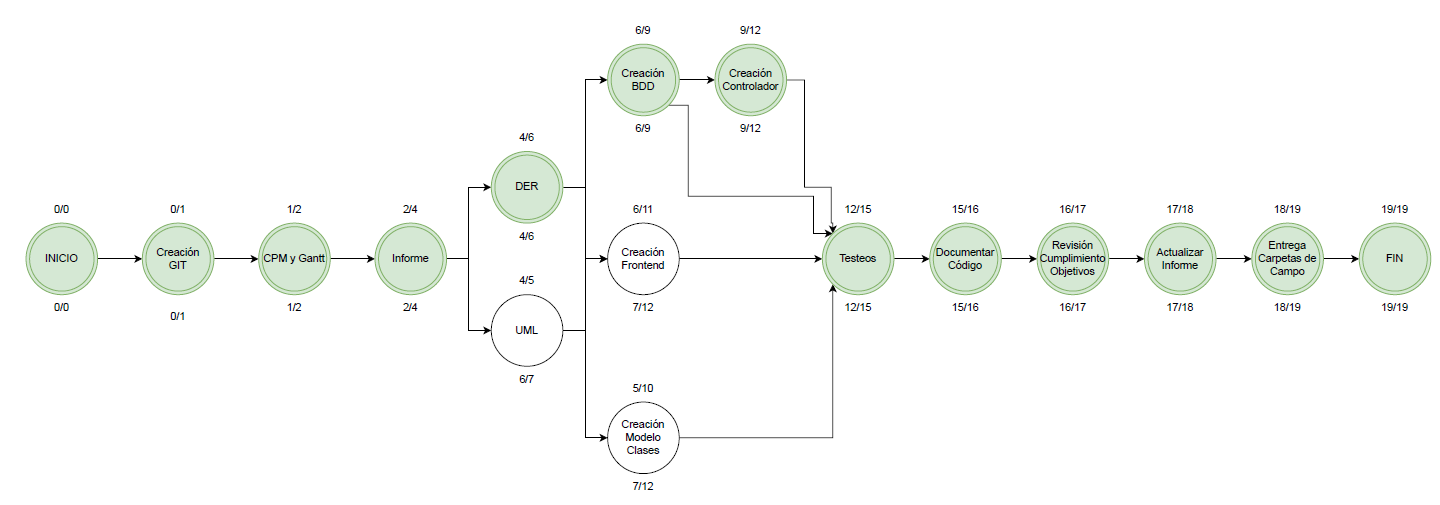
O también clonarlo en Git Bash con el siguiente código:

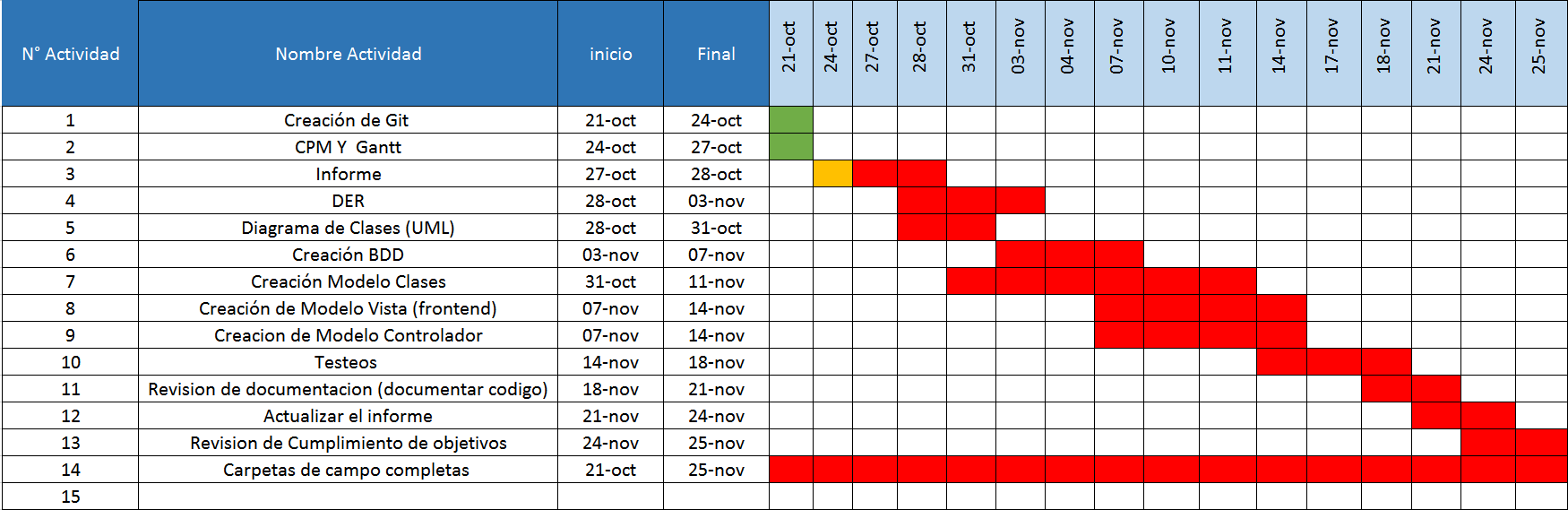
git clone <https://github.com/nietonicolasleon/Consultorio_G1.git>

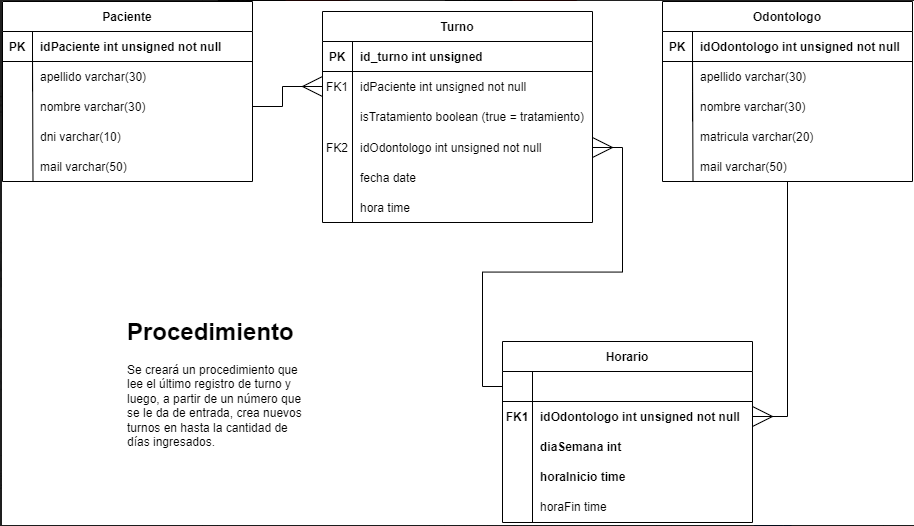
El mismo día, realizamos de manera grupal la Tabla de Tareas, el CPM, el Gantt y el DER, así como una versión muy temprana del Diagrama de Clases.

**Tabla de Tareas:**

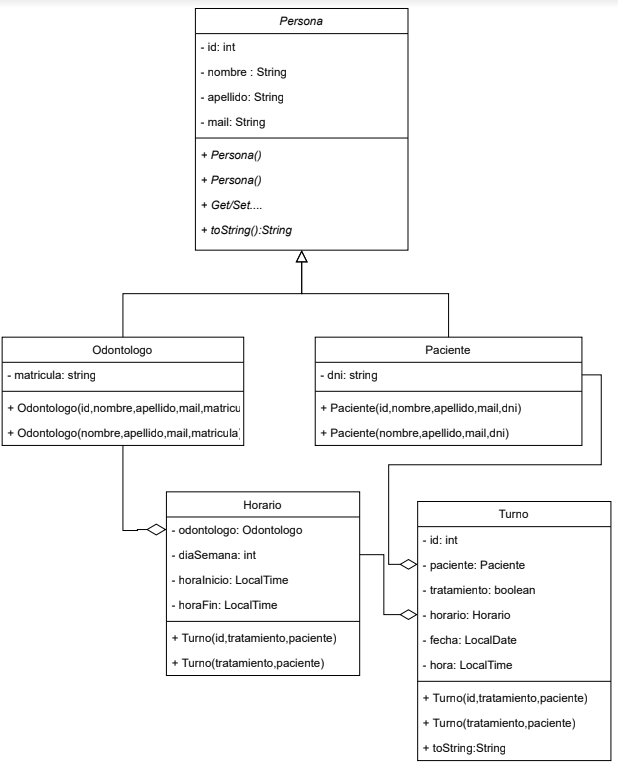


**CPM:**

**Gantt:**

**DER:**

**UML (versión temprana):**



24/10/22 **Informe**:

Hice el informe del proyecto. Es una descripción más extendida del mismo y se encuentra en la carpeta de Información del repositorio.

**Programación de la Base de Datos**:

Tras haber finalizado el informe, empecé a trabajar con la base de datos. La primera versión de la misma contaba con el procedimiento pero sin código funcional dentro, mientras que ya se habían insertado los datos de los odontólogos y sus horarios.

Código v1:

create database consultorio;

use consultorio;

create table odontologo(

idOdontologo int unsigned not null,

apellido varchar(30),

nombre varchar(30),

matricula varchar(20),

mail varchar(50),

primary key(idOdontologo)

);

create table paciente(

idPaciente int unsigned,

apellido varchar(30),

nombre varchar(30),

dni varchar(10),

mail varchar(50),

primary key(idPaciente)

);

create table horario(

idOdontologo int unsigned not null,

diaSemana int,

horaInicio time,

horaFin time,

primary key(idOdontologo, diaSemana, horaInicio),

foreign key(idOdontologo) references odontologo(idOdontologo)

);

create table turno(

idTurno int unsigned auto\_increment,

idPaciente int unsigned,

idOdontologo int unsigned not null,

isTratamiento boolean,

fecha date,

hora time,

primary key(idTurno),

foreign key(idPaciente) references paciente(idPaciente),

foreign key(idOdontologo) references horario(idOdontologo)

);

insert into odontologo values(1, "Vasquez", "Edmundo", "46343", "vasquezdrmundolol@mail.com");

insert into odontologo values(2, "Van Bothoven", "Ludwig", "27482", "drludwigvanbothoventf2@mail.com");

insert into odontologo values(3, "Taylor", "Roger", "78336", "drrogerqueentaylor@mail.com");

insert into odontologo values(4, "Ziegler", "Angela", "19216", "ziegleroangela@mail.com");

insert into odontologo values(5, "Valentine", "Vincent", "33844", "valentinevincent@mail.com");

insert into horario values(1, 1, "8:00", "12:30");

insert into horario values(1, 2, "8:00", "12:30");

insert into horario values(1, 3, "8:00", "12:30");

insert into horario values(2, 1, "9:30", "14:00");

insert into horario values(2, 3, "9:30", "14:00");

insert into horario values(2, 5, "9:30", "14:00");

insert into horario values(3, 2, "10:00", "15:00");

insert into horario values(3, 4, "10:00", "15:00");

insert into horario values(3, 6, "10:00", "15:00");

insert into horario values(4, 4, "8:00", "16:00");

insert into horario values(4, 5, "8:00", "16:00");

insert into horario values(5, 1, "10:00", "12:00");

insert into horario values(5, 2, "10:00", "12:00");

insert into horario values(5, 3, "10:00", "12:00");

insert into horario values(5, 4, "10:00", "12:00");

insert into horario values(5, 5, "10:00", "12:00");

delimiter $$

drop function if exists fechaSiguiente$$

create function fechaSiguiente(idO int unsigned)

returns date

begin

declare ultimoTurno date;

declare nextFecha date;

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

return nextFecha;

end$$

drop procedure if exists generarTurnos$$

create procedure generarTurnos(in cantDias int, in idO int unsigned)

begin

declare ultimoTurno date;

declare horaIni time;

declare durTurno time;

declare horaMax time;

declare contador int;

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

set contador = 0;

set durTurno = "0:30";

set horaIni = (select horaInicio from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(ultimoTurno, "%w"));

set horaMax = (select horaFin from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(ultimoTurno, "%w"));

while contador < cantDias do

/\*Llamar a la función fechaSiguiente\*/

while horaIni <= horaMax do

insert into turno(idOdontologo, isTratamiento, fecha, hora) values(idO, false, /\*RETURN DE fechaSiguiente\*/, horaIni);

set horaIni = addtime(horaIni, durTurno);

end while;

set contador = contador + 1;

end while;

end$$

delimiter ;

Más tarde decidí documentar el código

**Código v2**:

/\*Se crea y se usa la base de datos\*/

create database consultorio;

use consultorio;

/\*Se crean las tablas que serán utilizadas\*/

create table odontologo(

idOdontologo int unsigned not null,

apellido varchar(30),

nombre varchar(30),

matricula varchar(20),

mail varchar(50),

primary key(idOdontologo)

);

create table paciente(

idPaciente int unsigned,

apellido varchar(30),

nombre varchar(30),

dni varchar(10),

mail varchar(50),

primary key(idPaciente)

);

create table horario(

idOdontologo int unsigned not null,

diaSemana int,

horaInicio time,

horaFin time,

primary key(idOdontologo, diaSemana, horaInicio),

foreign key(idOdontologo) references odontologo(idOdontologo)

);

create table turno(

idTurno int unsigned auto\_increment,

idPaciente int unsigned,

idOdontologo int unsigned not null,

isTratamiento boolean,

fecha date,

hora time,

primary key(idTurno),

foreign key(idPaciente) references paciente(idPaciente),

foreign key(idOdontologo) references horario(idOdontologo)

);

/\*Se insertan los valores de los odontólogos y sus respectivos horarios\*/

insert into odontologo values(1, "Vasquez", "Edmundo", "46343", "vasquezdrmundolol@mail.com");

insert into odontologo values(2, "Van Bothoven", "Ludwig", "27482", "drludwigvanbothoventf2@mail.com");

insert into odontologo values(3, "Taylor", "Roger", "78336", "drrogerqueentaylor@mail.com");

insert into odontologo values(4, "Ziegler", "Angela", "19216", "ziegleroangela@mail.com");

insert into odontologo values(5, "Valentine", "Vincent", "33844", "valentinevincent@mail.com");

insert into horario values(1, 1, "8:00", "12:30");

insert into horario values(1, 2, "8:00", "12:30");

insert into horario values(1, 3, "8:00", "12:30");

insert into horario values(2, 1, "9:30", "14:00");

insert into horario values(2, 3, "9:30", "14:00");

insert into horario values(2, 5, "9:30", "14:00");

insert into horario values(3, 2, "10:00", "15:00");

insert into horario values(3, 4, "10:00", "15:00");

insert into horario values(3, 6, "10:00", "15:00");

insert into horario values(4, 4, "8:00", "16:00");

insert into horario values(4, 5, "8:00", "16:00");

insert into horario values(5, 1, "10:00", "12:00");

insert into horario values(5, 2, "10:00", "12:00");

insert into horario values(5, 3, "10:00", "12:00");

insert into horario values(5, 4, "10:00", "12:00");

insert into horario values(5, 5, "10:00", "12:00");

/\*Creamos las funciones y procedimientos a utilizar\*/

delimiter $$

/\*Esta función devuelve la siguiente fecha que corresponda a cada odontólogo, teniendo en cuenta cuáles son los días de semana en los que trabaja.\*/

drop function if exists fechaSiguiente$$

create function fechaSiguiente(idO int unsigned)

returns date

begin

declare ultimoTurno date;

declare nextFecha date;

declare diaUTurno int;

declare nextDia int;

/\*Se declara un handler por si no hay registros en la tabla turno\*/

declare continue handler for sqlstate '' set ultimoTurno = current\_date();

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

set diaUTurno = date\_format(ultimoTurno, "%w");

set nextFecha = adddate(ultimoTurno, )

return nextFecha;

end$$

/\*Este procedimiento genera turnos a partir de la última fecha. Requiere la cantidad de días que deseamos y el ID del odontólogo\*/

drop procedure if exists generarTurnos$$

create procedure generarTurnos(in cantDias int, in idO int unsigned)

begin

declare ultimoTurno date;

declare horaIni time;

declare durTurno time;

declare horaMax time;

declare contador int;

/\*Se declara un handler por si no hay registros en la tabla turno\*/

declare continue handler for sqlstate '' set ultimoTurno = current\_date();

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

set contador = 0;

set durTurno = "0:30";

set horaIni = (select horaInicio from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(ultimoTurno, "%w"));

set horaMax = (select horaFin from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(ultimoTurno, "%w"));

while contador < cantDias do

/\*Llamar a la función fechaSiguiente\*/

while horaIni <= horaMax do

insert into turno(idOdontologo, isTratamiento, fecha, hora) values(idO, false, /\*RETURN DE fechaSiguiente\*/, horaIni);

set horaIni = addtime(horaIni, durTurno);

end while;

set contador = contador + 1;

end while;

end$$

delimiter ;

28/11/22 Intenté varias formas de hacer funcionar la **función fechaSiguiente**, siendo esta una de las más cercanas a cumplir su propósito.

**Código v3 (solo la función):**

/\*Creamos las funciones y procedimientos a utilizar\*/

delimiter $$

/\*Esta función devuelve la siguiente fecha que corresponda a cada odontólogo, teniendo en cuenta cuáles son los días de semana en los que trabaja.\*/

drop function if exists fechaSiguiente$$

create function fechaSiguiente(idO int unsigned)

returns date

begin

declare ultimoTurno date;

declare nextFecha date;

declare diaUTurno int;

declare nextDia int;

/\*Se declara un handler por si no hay registros en la tabla turno\*/

declare continue handler for sqlstate '22002' set ultimoTurno = current\_date();

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

set diaUTurno = date\_format(ultimoTurno, "%w");

if diaUTurno = 6 then

set nextDia = (select min(diaSemana) from horario where horario.idOdontologo = idO);

else

set nextDia = (select min(diaSemana) from horario where diaSemana > diaUTurno and horario.idOdontologo = idO);

end if;

set nextFecha = adddate(ultimoTurno, interval nextDia day);

return nextFecha;

end$$

31/10/22 La profesora Cristina nos presentó la devolución del UML, pidiéndonos que agreguemos nuevas clases e implementemos una interface:

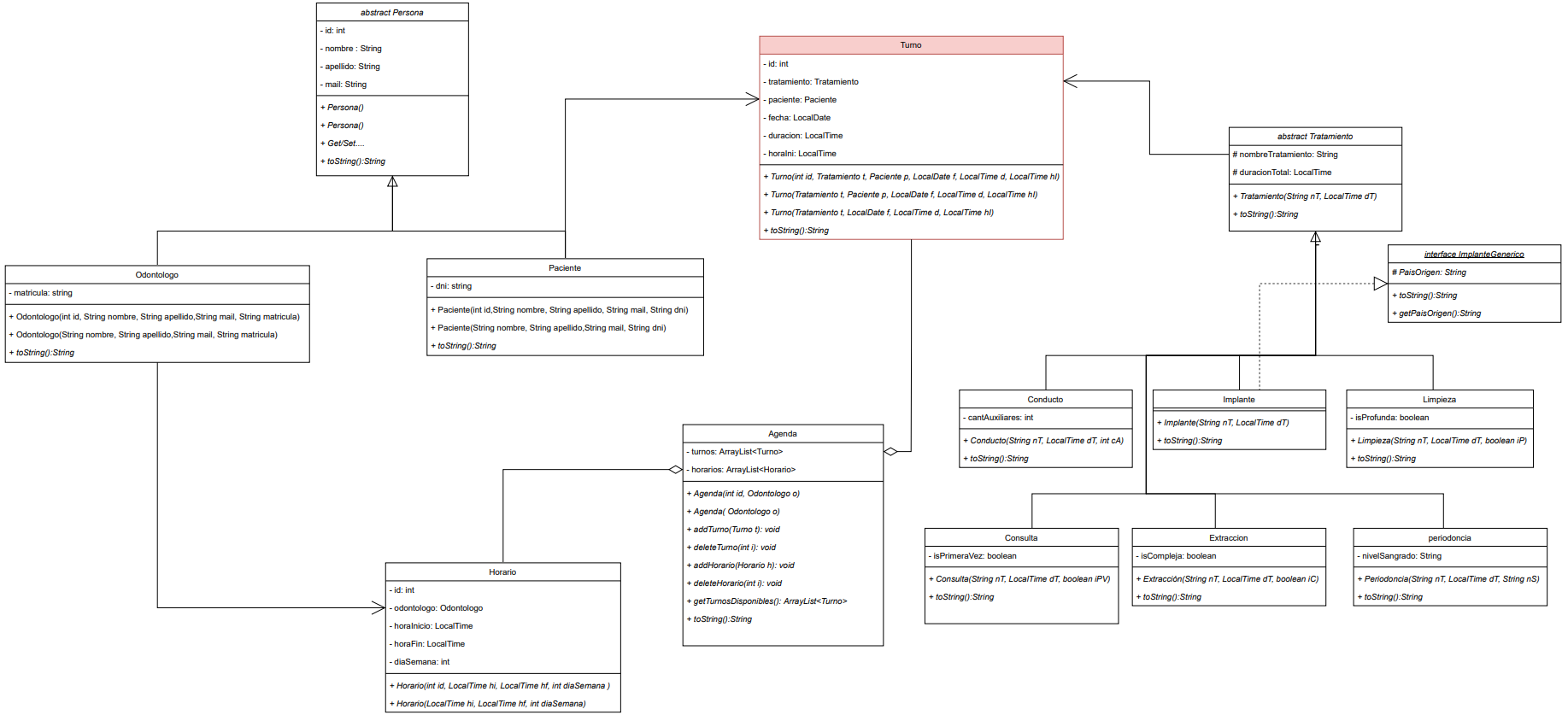
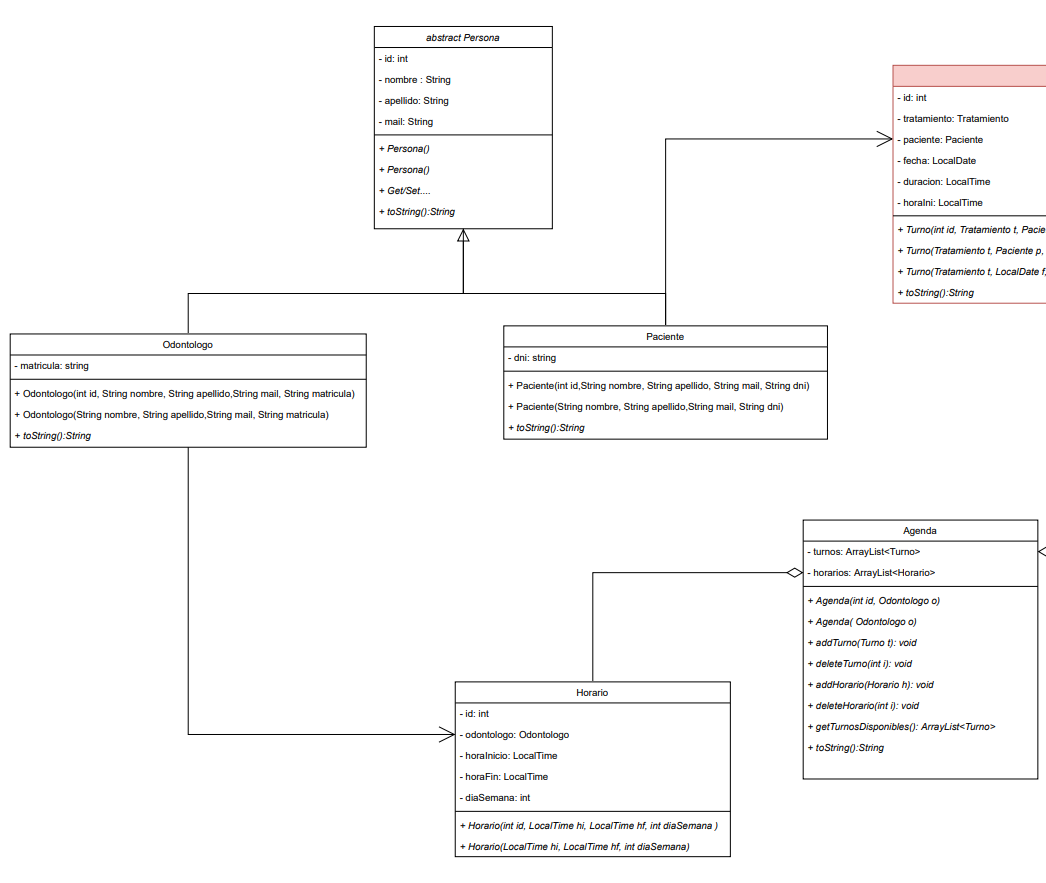
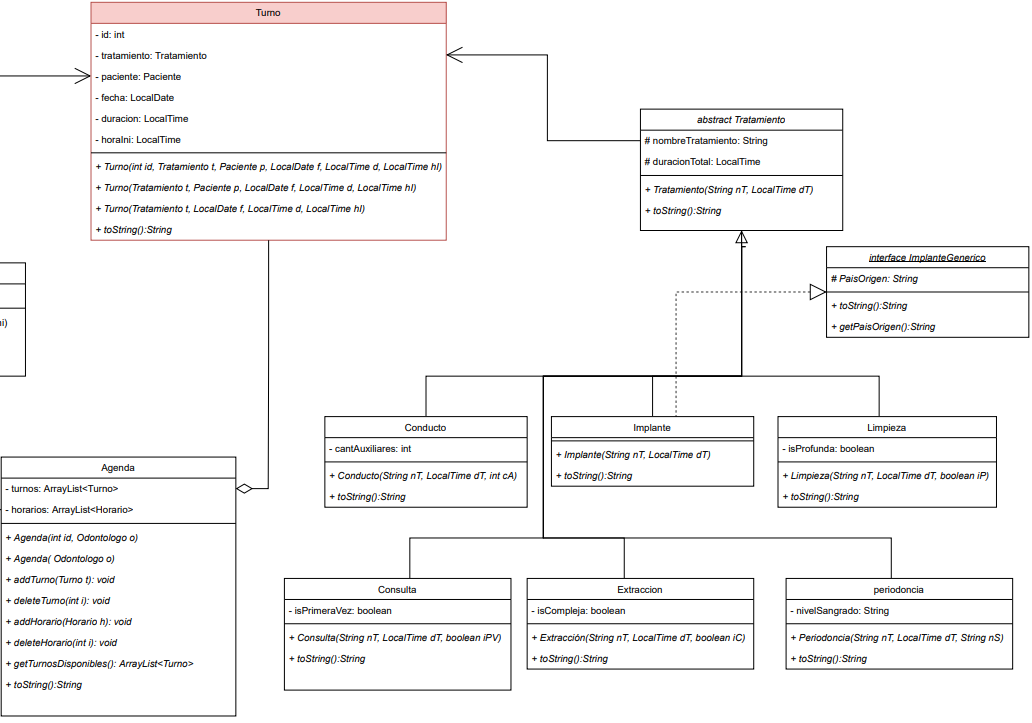


Imagen en dos partes:





3/11/22 Con ayuda de Dylan, **creamos las funciones para que el procedimiento haga efecto.**

Código v4 (funciones y procedimiento):

/\*Creamos las funciones y procedimientos a utilizar\*/

delimiter $$

/\*

Esta función busca el siguiente día laboral.

Toma por entrada el id del Odóntologo y el dia de la semana.

Si dicho día es uno donde el odontólogo trabaja, devuelve true.

\*/

drop function if exists isDiaLaborable $$

create function isDiaLaborable(idO int unsigned,diaIngresado int)

returns boolean

begin

declare done boolean default false;

declare isLaboral boolean;

declare nDia int;

declare cur1 cursor for select diaSemana from horario where idOdontologo = idO;

/\*Se declara un handler para verificar si el cursor terminó de recorrerse\*/

declare continue HANDLER for not found set done = true;

open cur1;

while done = false do

fetch cur1 into nDia;

if nDia = diaIngresado then

set isLaboral = true;

set done = true;

end if;

end while;

close cur1;

return isLaboral;

end$$

/\*Esta función devuelve la siguiente fecha que corresponda a cada odontólogo, teniendo en cuenta cuáles son los días de semana en los que trabaja.\*/

drop function if exists fechaSiguiente $$

create function fechaSiguiente(idO int unsigned, ultimoTurno date )

returns int

begin

declare nextFecha date;

declare diaUTurno int;

declare diasTranscuridos int default 0;

declare diaSemana int;

declare done boolean DEFAULT false;

set diaUTurno = date\_format(ultimoTurno, "%w");

set diaSemana = diaUTurno;

/\*

Se recorren todos los días hasta el primero donde le corresponda trabajar.

Luego se cuentan y se suman al últimoTurno para generar la nueva fecha.

\*/

while !done do

set diaSemana = diaSemana + 1;

set diasTranscuridos = diasTranscuridos + 1;

/\*Se resetea la semana tras el día 6 (sábado)\*/

if diaSemana > 6 then

set diaSemana = 0;

end if;

/\*Se llama a la función para comprobar el siguiente día de trabajo\*/

if isDiaLaborable(idO,diaSemana) then

set done = true;

end if;

end while;

set nextFecha = DATE\_ADD(ultimoTurno, INTERVAL diasTranscuridos DAY) ;

return nextFecha;

end$$

/\*Este procedimiento genera turnos a partir de la última fecha. Requiere la cantidad de días que deseamos y el ID del odontólogo\*/

drop procedure if exists generarTurnos$$

create procedure generarTurnos(in cantDias int, in idO int unsigned)

begin

declare ultimoTurno date;

declare fechaActual date;

declare horaIni time;

declare durTurno time;

declare horaMax time;

declare contador int;

set contador = 0;

set durTurno = "0:30";

while contador < cantDias do

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

if ultimoTurno is null then

set ultimoTurno = current\_date();

end if;

/\*Se llama a la función de fechaSiguiente para crear los turnos de cada día.\*/

set fechaActual = fechaSiguiente(idO, ultimoTurno);

set horaIni = (select horaInicio from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(fechaActual, "%w"));

set horaMax = (select horaFin from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(fechaActual, "%w"));

while horaIni < horaMax do

insert into turno(idOdontologo, isTratamiento, fecha, hora) values(idO, false, fechaActual, horaIni);

set horaIni = addtime(horaIni, durTurno);

end while;

set contador = contador + 1;

end while;

end$$

delimiter ;

11/11/22 Pasé algunos datos de la rama de Dylan a la mía y probé **crear objetos Odontólogo en Java con datos tomados de la BDD** en el paquete datos:

package datos;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.Statement;

import java.sql.SQLException;

import java.sql.ResultSet;

import java.time.LocalDate;

import java.time.LocalTime;

import java.util.ArrayList;

import java.time.LocalTime;

import modelo.\*;

public class accesoDatos {

private Connection conn;

private Statement comandos;

private String url = "jdbc:mysql://localhost/consultorio";

private String user = "root";

private String password = "";

public Limpieza l1 = new Limpieza("Limpieza tranqui", LocalTime.of(1, 0), false);

public Agenda a1 = new Agenda();

public Agenda a2 = new Agenda();

@@ -22,9 +27,39 @@ public class accesoDatos {

public Agenda a5 = new Agenda();

public accesoDatos(){

try{

Class.forName("com.mysql.cj.jdbc.Driver");

conn = DriverManager.getConnection(url, user, password);

comandos = conn.createStatement();

} catch(ClassNotFoundException ex){

System.out.println("No se encontro la BDD.");

} catch (SQLException ex){

System.out.println("Usuario o password incorrectos.");

}

}

public ArrayList<Odontologo> getOdontologos(){

ArrayList<Odontologo> ods = new ArrayList();

String sql = "select \* from odontologo";

try{

ResultSet tabla = comandos.executeQuery(sql);

while(tabla.next()){

int id = tabla.getInt("idOdontologo");

String apellido = tabla.getString("apellido");

String nombre = tabla.getString("nombre");

String matricula = tabla.getString("matricula");

String mail = tabla.getString("mail");

Odontologo o = new Odontologo(id, nombre, apellido, mail, matricula);

ods.add(o);

}

}catch(SQLException ex){

System.out.println("ERROR:" + ex.toString());

}

return ods;

}

}

14/11/22 Tras una charla con el grupo sobre el estado del proyecto y de la base de datos, **Agregué la tabla Tratamiento**

**Código v5 (tabla e inserts):**

create table tratamiento(

idTratamiento int unsigned not null,

nombre varchar(30),

duracion time,

descripcion varchar(100),

primary key(idTratamiento)

);

insert into tratamiento values(1, "Consulta Primera vez", "0:30", "true");

insert into tratamiento values(2, "Consulta", "0:30", "false");

insert into tratamiento values(3, "Conducto 1 auxiliar", "1:00", "1");

insert into tratamiento values(4, "Conducto 2 auxiliares", "1:00", "2");

insert into tratamiento values(5, "Limpieza Simple", "0:30", "false");

insert into tratamiento values(6, "Limpieza Profunda", "0:30", "true");

insert into tratamiento values(7, "Implante Chino", "2:00", "China");

insert into tratamiento values(8, "Implante Ruso", "2:00", "Rusia");

insert into tratamiento values(9, "Extraccion Simple", "2:00", "false");

insert into tratamiento values(10, "Extraccion Compleja", "2:00", "true");

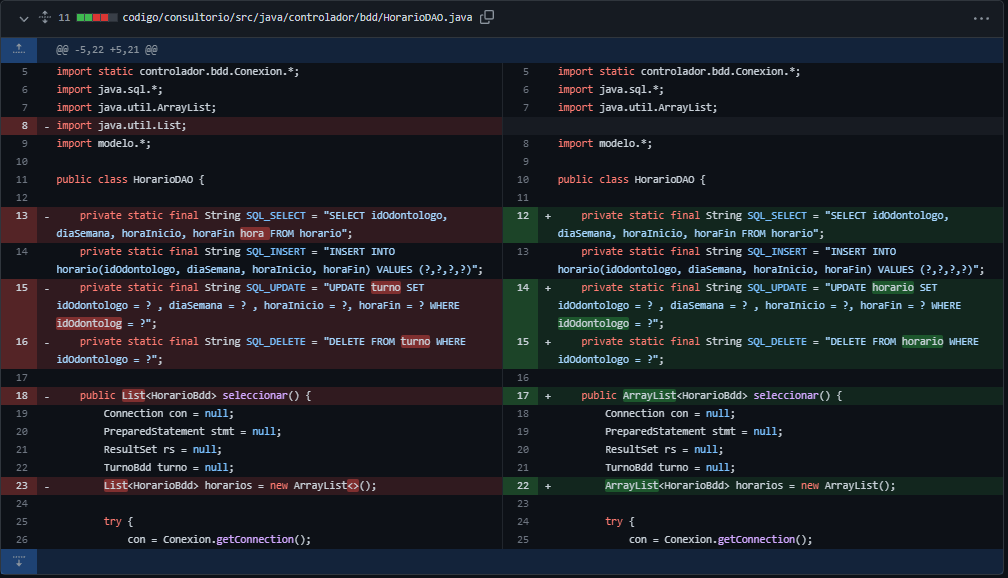
insert into tratamiento values(11, "Periodoncia Simple", "1:00", "Bajo");

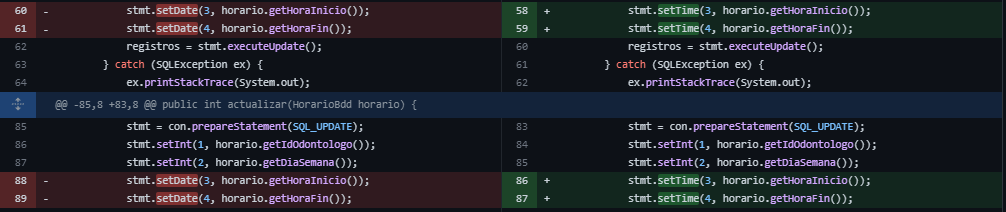
insert into tratamiento values(12, "Periodoncia Compleja", "1:00", "Alto");

También, el mismo día empecé a trabajar y dar una mano en el Controlador, puesto que veníamos atrasados en ese aspecto.

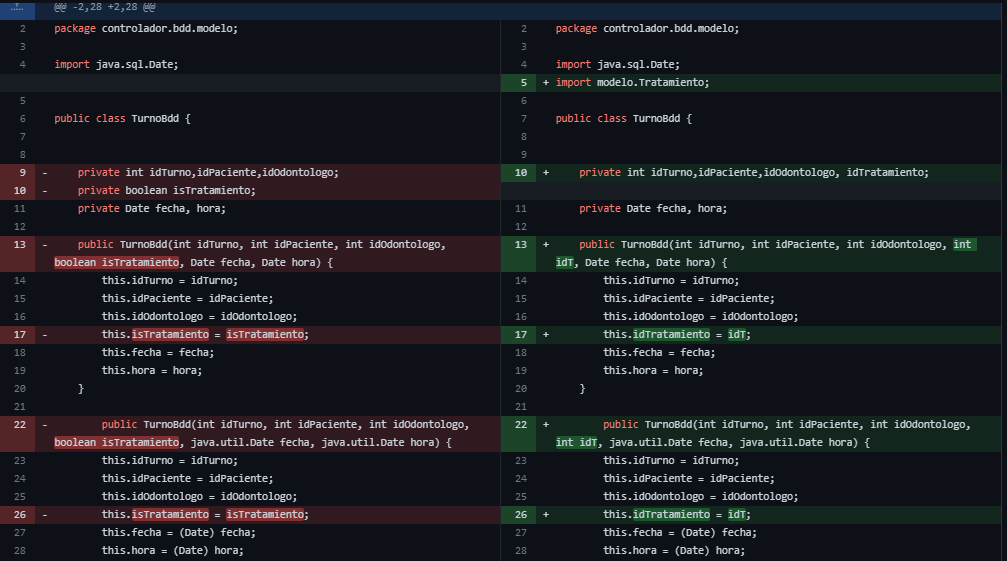
Me dediqué a que las cosas que cambié en la BDD encajasen en el proyecto y corregir algunos errores. Adjunto algunas fotos como ejemplo:

Corrección de errores: Se arreglaron algunos errores de typeo y se empleó el uso de las ArrayList; así como el cambio de dato del mal implementado DATE en Java a TIME (el que se debía usar).





Modificación para que el proyecto encaje con la BDD: Se cambió la variable boolean isTratamiento a la variable integer idTratamiento (BDD) y se arreglaron los inserts para que funciones



También era necesario algo que permita acceder a la tabla de Tratamientos y poder seleccionar o modificar sus datos, por lo que creé las clases TratamientoDAO (para acceder a los datos) y TratamientoBDD (puesto que eran distintos los tratamientos de la BDD que los hijos creados en el modelo):

**TratamientoDAO:**

package controlador.bdd;

import java.util.Date;

import static controlador.bdd.Conexion.\*;

import java.sql.\*;

import java.util.ArrayList;

import controlador.bdd.modelo.TratamientoBDD;

public class TratamientoDAO {

private static final String SQL\_SELECT = "SELECT idTratamiento, nombre, duracion, datos FROM tratamiento";

private static final String SQL\_INSERT = "INSERT INTO tratamiento(idTratamiento, nombre, duracion, datos) VALUES (?,?,?,?)";

private static final String SQL\_UPDATE = "UPDATE tratamiento SET idTratamiento = ? , nombre = ? , duracion = ?, datos = ? WHERE idTratamiento = ?";

private static final String SQL\_DELETE = "DELETE FROM tratamiento WHERE idTratamiento = ?";

public ArrayList<TratamientoBDD> seleccionar() {

Connection con = null;

PreparedStatement stmt = null;

ResultSet rs = null;

ArrayList<TratamientoBDD> tratamientos = new ArrayList();

try {

con = Conexion.getConnection();

stmt = con.prepareStatement(SQL\_SELECT);

rs = stmt.executeQuery();

while (rs.next()) {

int idTratamiento = rs.getInt("idTratamiento");

String nombre = rs.getString("nombre");

Time duracion = rs.getTime("duracion");

String datos = rs.getString("datos");

TratamientoBDD tratamiento = new TratamientoBDD(idTratamiento, nombre, duracion, datos);

tratamientos.add(tratamiento);

}

} catch (SQLException ex) {

ex.printStackTrace(System.out);

} finally {

try {

close(stmt);

close(rs);

close(con);

} catch (SQLException ex) {

ex.printStackTrace(System.out);

}

}

return tratamientos;

}

public int insertar(TratamientoBDD tratamiento) {

Connection con = null;

PreparedStatement stmt = null;

int registros = 0;

try {

con = getConnection();

stmt = con.prepareStatement(SQL\_INSERT);

stmt.setInt(1, tratamiento.getIdTratamiento());

stmt.setString(2, tratamiento.getNombre());

stmt.setTime(3, tratamiento.getDuracion());

stmt.setString(4, tratamiento.getDatos());

registros = stmt.executeUpdate();

} catch (SQLException ex) {

ex.printStackTrace(System.out);

} finally {

try {

close(stmt);

close(con);

} catch (SQLException ex) {

ex.printStackTrace(System.out);

}

}

return registros;

}

public int actualizar(TratamientoBDD tratamiento) {

Connection con = null;

PreparedStatement stmt = null;

int registros = 0;

try {

con = getConnection();

stmt = con.prepareStatement(SQL\_UPDATE);

stmt.setInt(1, tratamiento.getIdTratamiento());

stmt.setString(2, tratamiento.getNombre());

stmt.setTime(3, tratamiento.getDuracion());

stmt.setString(4, tratamiento.getDatos());

registros = stmt.executeUpdate();

} catch (SQLException ex) {

ex.printStackTrace(System.out);

} finally {

try {

close(stmt);

close(con);

} catch (SQLException ex) {

ex.printStackTrace(System.out);

}

}

return registros;

}

public int eliminar(TratamientoBDD tratamiento) {

Connection con = null;

PreparedStatement stmt = null;

int registros = 0;

try {

con = getConnection();

stmt = con.prepareStatement(SQL\_DELETE);

stmt.setInt(1, tratamiento.getIdTratamiento());

registros = stmt.executeUpdate();

} catch (SQLException ex) {

ex.printStackTrace(System.out);

} finally {

try {

close(stmt);

close(con);

} catch (SQLException ex) {

ex.printStackTrace(System.out);

}

}

return registros;

}

}

**TratamientoBDD:**

package controlador.bdd.modelo;

import java.sql.Time;

public class TratamientoBDD {

private int idTratamiento;

private Time duracion;

private String nombre, datos;

public TratamientoBDD(int id, String n, Time dur, String da){

this.idTratamiento = id;

this.nombre = n;

this.duracion = dur;

this.datos = da;

}

public int getIdTratamiento() {

return idTratamiento;

}

public void setIdTratamiento(int idTratamiento) {

this.idTratamiento = idTratamiento;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public Time getDuracion() {

return duracion;

}

public void setDuracion(Time duracion) {

this.duracion = duracion;

}

public String getDatos() {

return datos;

}

public void setDatos(String datos) {

this.datos = datos;

}

}

17/11/22 **Finalmente, así quedó la versión final de la BDD.**

**Código final:**

/\*Se crea y se usa la base de datos\*/

drop database if exists consultorio;

create database consultorio;

use consultorio;

/\*Se crean las tablas que serán utilizadas\*/

create table odontologo(

idOdontologo int unsigned auto\_increment not null,

apellido varchar(30),

nombre varchar(30),

matricula varchar(20) unique,

mail varchar(50),

primary key(idOdontologo)

);

create table paciente(

idPaciente int unsigned auto\_increment not null,

apellido varchar(30),

nombre varchar(30),

dni varchar(10) unique,

mail varchar(50),

primary key(idPaciente)

);

create table horario(

idOdontologo int unsigned not null,

diaSemana int,

horaInicio time,

horaFin time,

primary key(idOdontologo, diaSemana, horaInicio),

foreign key(idOdontologo) references odontologo(idOdontologo)

);

create table tratamiento(

idTratamiento int unsigned not null,

nombre varchar(30),

duracion time,

datos varchar(50),

primary key(idTratamiento)

);

create table turno(

idTurno int unsigned auto\_increment not null,

idPaciente int unsigned,

idOdontologo int unsigned not null,

idTratamiento int unsigned,

fecha date,

hora time,

primary key(idTurno),

foreign key(idPaciente) references paciente(idPaciente),

foreign key(idOdontologo) references horario(idOdontologo),

foreign key(idTratamiento) references tratamiento(idTratamiento)

);

/\*Se insertan los valores de los odontólogos y sus respectivos horarios\*/

insert into odontologo values(1, "Vasquez", "Edmundo", "46343", "vasquezdrmundolol@mail.com");

insert into odontologo values(2, "Van Bothoven", "Ludwig", "27482", "drludwigvanbothoventf2@mail.com");

insert into odontologo values(3, "Taylor", "Roger", "78336", "drrogerqueentaylor@mail.com");

insert into odontologo values(4, "Ziegler", "Angela", "19216", "ziegleroangela@mail.com");

insert into odontologo values(5, "Valentine", "Vincent", "33844", "valentinevincent@mail.com");

insert into horario values(1, 1, "8:00", "12:30");

insert into horario values(1, 2, "8:00", "12:30");

insert into horario values(1, 3, "8:00", "12:30");

insert into horario values(2, 1, "9:30", "14:00");

insert into horario values(2, 3, "9:30", "14:00");

insert into horario values(2, 5, "9:30", "14:00");

insert into horario values(3, 2, "10:00", "15:00");

insert into horario values(3, 4, "10:00", "15:00");

insert into horario values(3, 6, "10:00", "15:00");

insert into horario values(4, 4, "8:00", "16:00");

insert into horario values(4, 5, "8:00", "16:00");

insert into horario values(5, 1, "10:00", "12:00");

insert into horario values(5, 2, "10:00", "12:00");

insert into horario values(5, 3, "10:00", "12:00");

insert into horario values(5, 4, "10:00", "12:00");

insert into horario values(5, 5, "10:00", "12:00");

insert into tratamiento values(1, "Consulta Primera vez", "0:30", "true");

insert into tratamiento values(2, "Consulta", "0:30", "false");

insert into tratamiento values(3, "Conducto 1 auxiliar", "1:00", "1");

insert into tratamiento values(4, "Conducto 2 auxiliares", "1:00", "2");

insert into tratamiento values(5, "Limpieza Simple", "0:30", "false");

insert into tratamiento values(6, "Limpieza Profunda", "0:30", "true");

insert into tratamiento values(7, "Implante Chino", "2:00", "China");

insert into tratamiento values(8, "Implante Ruso", "2:00", "Rusia");

insert into tratamiento values(9, "Extraccion Simple", "2:00", "false");

insert into tratamiento values(10, "Extraccion Compleja", "2:00", "true");

insert into tratamiento values(11, "Periodoncia Simple", "1:00", "Bajo");

insert into tratamiento values(12, "Periodoncia Compleja", "1:00", "Alto");

/\*Creamos las funciones y procedimientos a utilizar\*/

delimiter $$

/\*

Esta función busca el siguiente día laboral.

Toma por entrada el id del Odóntologo y el dia de la semana.

Si dicho día es uno donde el odontólogo trabaja, devuelve true.

\*/

drop function if exists isDiaLaborable $$

create function isDiaLaborable(idO int unsigned,diaIngresado int)

returns boolean

begin

declare done boolean default false;

declare isLaboral boolean;

declare nDia int;

declare cur1 cursor for select diaSemana from horario where idOdontologo = idO;

/\*Se declara un handler para verificar si el cursor terminó de recorrerse\*/

declare continue HANDLER for not found set done = true;

open cur1;

while done = false do

fetch cur1 into nDia;

if nDia = diaIngresado then

set isLaboral = true;

set done = true;

end if;

end while;

close cur1;

return isLaboral;

end$$

/\*Esta función devuelve la siguiente fecha que corresponda a cada odontólogo, teniendo en cuenta cuáles son los días de semana en los que trabaja.\*/

drop function if exists fechaSiguiente $$

create function fechaSiguiente(idO int unsigned,ultimoTurno date )

returns int

begin

declare nextFecha date;

declare diaUTurno int;

declare diasTranscuridos int default 0;

declare diaSemana int;

declare done boolean DEFAULT false;

set diaUTurno = date\_format(ultimoTurno, "%w");

set diaSemana = diaUTurno;

/\*

Se recorren todos los días hasta el primero donde le corresponda trabajar.

Luego se cuentan y se suman al últimoTurno para generar la nueva fecha.

\*/

while !done do

set diaSemana = diaSemana + 1;

set diasTranscuridos = diasTranscuridos + 1;

/\*Se resetea la semana tras el día 6 (sábado)\*/

if diaSemana > 6 then

set diaSemana = 0;

end if;

/\*Se llama a la función para comprobar el siguiente día de trabajo\*/

if isDiaLaborable(idO,diaSemana) then

set done = true;

end if;

end while;

set nextFecha = DATE\_ADD(ultimoTurno, INTERVAL diasTranscuridos DAY) ;

return nextFecha;

end$$

/\*Este procedimiento genera turnos a partir de la última fecha. Requiere la cantidad de días que deseamos y el ID del odontólogo\*/

drop procedure if exists generarTurnos$$

create procedure generarTurnos(in cantDias int, in idO int unsigned)

begin

declare ultimoTurno date;

declare fechaActual date;

declare horaIni time;

declare durTurno time;

declare horaMax time;

declare contador int;

set contador = 0;

set durTurno = "0:30";

while contador < cantDias do

set ultimoTurno = (select max(fecha) from turno where turno.idOdontologo = idO);

if ultimoTurno is null then

set ultimoTurno = current\_date();

end if;

/\*Se llama a la función de fechaSiguiente para crear los turnos de cada día.\*/

set fechaActual = fechaSiguiente(idO, ultimoTurno);

set horaIni = (select horaInicio from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(fechaActual, "%w"));

set horaMax = (select horaFin from horario where horario.idOdontologo = idO and horario.diaSemana = date\_format(fechaActual, "%w"));

while horaIni < horaMax do

insert into turno(idOdontologo, fecha, hora) values(idO, fechaActual, horaIni);

set horaIni = addtime(horaIni, durTurno);

end while;

set contador = contador + 1;

end while;

end$$

delimiter ;

Más tarde, decidí continuar con el trabajo pendiente de **TratamientoDAO**. Ahora, el método **seleccionar()** no nos debía devolver una ArrayList de objetos TratamientoBDD, si no convertirlos a hijos de Tratamiento. **Para eso, implementé un for y que los 12 tratamientos se insertasen en una nueva lista**:

ArrayList<Tratamiento> modelTrats = new ArrayList();

for (int i = 0; i < tratamientos.size(); i++) {

switch(tratamientos.get(i).getNombre()){

case "Consulta Primera vez":

Consulta cpv = new Consulta(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), true);

modelTrats.add(cpv);

break;

case "Consulta":

Consulta c = new Consulta(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), false);

modelTrats.add(c);

break;

case "Conducto 1 auxiliar":

Conducto c1 = new Conducto(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), 1);

modelTrats.add(c1);

break;

case "Conducto 2 auxiliares":

Conducto c2 = new Conducto(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), 2);

modelTrats.add(c2);

break;

case "Limpieza Simple":

Limpieza ls = new Limpieza(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), false);

modelTrats.add(ls);

break;

case "Limpieza Profunda":

Limpieza lp = new Limpieza(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), true);

modelTrats.add(lp);

break;

case "Implante Chino":

Implante ip = new Implante(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), "China");

modelTrats.add(ip);

break;

case "Implante Ruso":

Implante ir = new Implante(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), "Rusia");

modelTrats.add(ir);

break;

case "Extraccion Simple":

Extraccion es = new Extraccion(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), false);

modelTrats.add(es);

break;

case "Extraccion Compleja":

Extraccion ec = new Extraccion(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), true);

modelTrats.add(ec);

break;

case "Periodoncia Simple":

Periodoncia ps = new Periodoncia(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), "Bajo");

modelTrats.add(ps);

break;

case "Periodoncia Compleja":

Periodoncia pc = new Periodoncia(tratamientos.get(i).getIdTratamiento(), tratamientos.get(i).getNombre(), tratamientos.get(i).getDuracion().toLocalTime(), "Alto");

modelTrats.add(pc);

break;

default:

System.out.println("Error.");

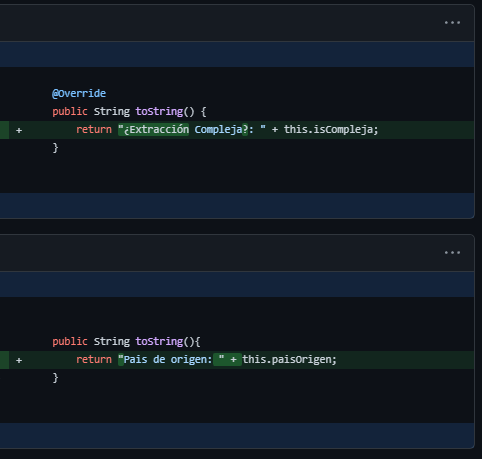
break;

}

}

return modelTrats;

Por último, corregí un algunos errores de typeo en los toString() de Tratamiento y sus hijas. Aquí algunas fotos de ejemplo:



Y por fin, tras mucho código, charlas con el grupo y los profesores, más código, documentación, y muchas, pero muchas correcciones tanto en los diagramas como en el código, el proyecto estaba listo.

O bueno, no del todo. Para la fecha de hoy, 17/11/2022, aún quedan algunas funciones para implementar, como la reserva y la filtración de turnos.

Y para las 21:45, la Carpeta de Campo de Nieto Nicolás 5° 1ra C, **se subió al GitHub**.