UNIVERSIDAD TECNOLÓGICA DE EL SALVADOR FACULTAD DE INFORMÁTICA Y CIENCIAS APLICADAS ESCUELA DE INFORMÀTICA				
Asignatura: Análisis y Diseño de Sistemas de Información para Bases de Datos **Docente:** Ing. Kirio Marvin Ventura Fuentes **Evaluación:** Parcial #3	**Sección 01**	**Ciclo 02-2020**	**Aula Virtual**	**Nota obtenida**
Alumno: De Paz Castellanos, Jocelyn Stefany			**No de carnet: 65-0994-2018**	
Alumno: Martínez Miranda, Rafael Antonio			**No de carnet: 65-4016-2018**	
Carrera: Ingeniería en gestión de bases de datos			**Fecha: 19/10/2020**	

Verificar si los estudiantes son capaces de:	SISTEMA DE EVALUACION		
Definir los sistemas de información para bases de datos.	Avance 2 Proyecto	40%	
	Parcial	60%	
	NOTA FINAL DE PERIODO		

INDICACIONES GENERALES:

- Primero lea todo el test, asegúrese de que esté completo, luego comience a contestarlo.
- Escriba claro y con tinta, toda corrección invalida su respuesta.
- No está permitido el acceso a ningún material de apoyo (impreso o digital), de incumplir esta disposición, la evaluación será anulada y la nota que se asignará será **cero puntos cero** (0.0).
- Por ningún motivo está permitido el uso de celular, cualquier incumplimiento anula la prueba. **NO SE ARRIESGUE.**

Desarrollo:

PRIMERA PARTE (Prueba de conocimientos) …… Ponderación 40%.

Indicaciones – Desarrolle correctamente el siguiente proceso de datos:

Crear una base de datos de planilla y crear los procedimientos almacenados para crear empleados y para calcular el pago de empleados de forma mensual.

Tablas utilizadas en la BD:

```
create table departamento(
id_departamento int not null identity(1,1) primary key,
nombre_departamento varchar(60) not null);

create table subdepartamento(
id_subdepartamento int not null identity(1,1) primary key,
nombre_subdepartamento varchar(60) not null,
id_departamento int not null,
constraint fk_depa_subdepa foreign key (id_departamento) references departamento(id_departamento));

create table sexo(
id_sexo int identity(1,1) primary key not null,
sexo varchar(20) not null
);
```

```sql
create table empleados(
id_empleado int not null identity(1,1) primary key,
nombres varchar(100) not null,
apellidos varchar(100) not null,
fecha_nacimiento date not null,
correo_electronico varchar(60) not null,
dui varchar(12) not null unique,
nit varchar(20),
fecha_registro date default(getdate()),
telefono varchar(10) not null,
estado bit default 1,
direccion varchar(250),
id_departamento int not null,
id_sexo int not null,
constraint fk_sexo_empleado foreign key (id_sexo)references sexo(id_sexo),
constraint fk_empleado_depa foreign key (id_departamento) references subdepartamento(id_subdepartamento));

create table plaza(
id_plaza int not null identity(1,1) primary key,
plaza varchar(60) not null);

create table turno_horario(
id_turno int not null identity(1,1) primary key,
turno varchar(60) not null);

create table contrato(
id_contrato int not null identity(1,1) primary key,
id_empleado int not null unique,
id_plaza int not null,
id_turno int not null,
horas_semana int not null,
salario_base real not null,
fecha_contrato date default(getdate()),
estado bit default (1),
constraint fk_plaza_contrato foreign key(id_plaza) references plaza(id_plaza),
constraint fk_empleado_contrato foreign key(id_empleado) references empleados (id_empleado),
constraint fk_turno_contrato foreign key (id_turno) references turno_horario(id_turno));
insert into contrato (id_empleado,id_plaza,id_turno,horas_semana,salario_base) values(1,1,1,40,500);
insert into contrato (id_empleado,id_plaza,id_turno,horas_semana,salario_base) values(2,1,2,35,500);

create table mes(
id_mes int not null identity(1,1) primary key,
mes varchar(30) not null);

create table asistencia_mensual(
id_asistencia int not null identity(1,1) primary key,
id_empleado int not null,
id_mes int not null,
fecha_creacion date default(getdate()),
dias_asistidos int not null,
total_horas_realizadas decimal(3) not null,
constraint fk_empleado_asistencia foreign key(id_empleado) references empleados(id_empleado),
constraint fk_mes_asistencia foreign key(id_mes) references mes(id_mes));

create table boleta_pago_mensual(
id_boleta int not null identity(1,1) primary key,
id_asistencia int not null,
id_mes int not null,
total_desc real not null,
salario_neto real not null,
fecha_generacion date default(getdate()),
constraint fk_mes_boleta foreign key(id_mes) references mes(id_mes),
constraint fk_asistencia_boleta foreign key(id_asistencia) references asistencia_mensual(id_asistencia));
```

```sql
create table tipo_movimiento(
id_movimiento int not null identity(1,1) primary key,
movimiento varchar(60) not null);

create table detalle_boleta(
id_detalle int not null identity(1,1) primary key,
id_boleta int not null,
id_mov int not null,
concepto varchar(60) not null,
monto real not null,
fecha_detalle date default(getdate()),
constraint fk_movimiento_detalle foreign key(id_mov) references tipo_movimiento(id_movimiento),
constraint fk_detalle_boleta foreign key(id_boleta) references boleta_pago_mensual(id_boleta));

create table tipo_descuento_ley(
id_tipo_des int primary key identity(1,1) not null,
nombre_desc varchar(60) not null
);

create table descuento_ley_detalle(
id_detalle_des int primary key identity(1,1) not null,
id_tipo_des int not null,
fecha_registro datetime default(getdate()),
porcentaje real not null,
categoria tinyint,
posee_rango bit not null,
vigencia bit not null,
constraint fk_tipo_descuento foreign key (id_tipo_des) references tipo_descuento_ley(id_tipo_des)
);

create table rango_descuento(
id_rango_descuento int not null primary key identity(1,1),
rango_inicial real not null,
rango_final real,
cuota real not null,
id_detalle_des int not null,
constraint fk_rango_descuento foreign key(id_detalle_des) references descuento_ley_detalle(id_detalle_des)
);
```

Procedimientos almacenados:

Ya que el procedimiento pa_boleta_pago es bastante extenso, se detallará lo que hace utilizando capturas de cada procedimiento interno

```sql
go
create procedure pa_boleta_pago...
```

Creación del procedimiento:

1. Se definen los parámetros de entrada
2. Se declaran variables que se utilizarán en los cálculos posteriores
3. Se inserta la asistencia mensual del empleado
4. Se calculan valores que se utilizarán más adelante, tales como el total de horas que debe cumplir un empleado en base a su contrato, o el precio de la hora basado en el salario base y el total de las horas

En esta parte, se recuperan valores que se necesitan para poder determinar cuanto de ese salario base fue devengado, a través de las horas mensuales que haya realizado el empleado en cuestión. A través de esto, se puede determinar si realizó horas extras, si terminó debiendo horas extras, y cuando es el abono/descuento

```
go
create procedure pa_boleta_pago

@pa_dias int,
@pa_horas real,
@pa_empleado int,
@pa_mes int

as
begin
```

```
declare @horario int, @salario_base real, @calculo_salario real, @precio_hora real;
declare @total_horas real = 0.0, @descuento real = 0.0, @horas_fuera real=0.0, @horas_extras real=0.0, @asis int=0;
declare @afp real=0.0, @isss real=0.0, @renta real=0.0, @sueldo real=0.0;
declare @descuento_total real=0.0, @salario_neto real=0.0, @descuento_renta real=0.0, @descuentos_total real=0.0;
```

```
insert into asistencia_mensual(dias_asistidos, total_horas_realizadas, id_empleado, id_mes)values(@pa_dias, @pa_horas, @pa_empleado, @pa_mes);

set @asis = (select (id_asistencia) from asistencia_mensual where id_empleado = @pa_empleado and id_asistencia = @@IDENTITY);
set @horario = (select id_turno from contrato where id_empleado = @pa_empleado);
set @salario_base = (select salario_base from contrato where id_empleado = @pa_empleado);
set @total_horas = (select (horas_semana) from contrato where id_empleado = @pa_empleado) * 4;
set @precio_hora = @salario_base/@total_horas;
```

Calculo del sueldo devengado a través del siguiente if dentro del mismo procedure

El primer filtro es si es turno matutino o nocturno (ya que ambos se pagan diferente). El segundo filtro, es evaluar si las horas que realizó el empleado se completaron o no fueron completadas.

Una vez determinado si fueron o no cumplidas, se procede con el descuento/abono y finalmente, se calcula el salario devengado, resultado guardado en la variable @calculo_salario

```
if @horario = 1
    begin
        if @pa_horas >= @total_horas
            begin
                set @calculo_salario = @precio_hora * @pa_horas;
                set @descuento = 0.0;
                set @horas_extras = ((@pa_horas - @total_horas) * @precio_hora) * 2;
            end
        else
            begin
                set @horas_extras = 0.0;
                set @horas_fuera = @total_horas - @pa_horas;
                set @descuento = @horas_fuera * @precio_hora;
                set @calculo_salario = (@precio_hora * @pa_horas) - @descuento;
            end
    end
else if @horario = 2
begin
    set @precio_hora = @precio_hora * 1.25;

        if @pa_horas >= @total_horas
            begin
                set @calculo_salario = @precio_hora * @pa_horas;
                set @descuento = 0.0;
                set @horas_extras = ((@pa_horas - @total_horas) * @precio_hora) * 2;
            end
        else if @pa_horas < @total_horas
            begin
                set @horas_extras = 0.0;
                set @horas_fuera = @total_horas - @pa_horas;
                set @descuento = @horas_fuera * @precio_hora;
                set @calculo_salario = (@precio_hora * @pa_horas) - @descuento;
            end
end
```

Calculo de los descuentos de ley AFP e ISSS

```
set @afp = (select top(1) (porcentaje) from descuento_ley_detalle where id_tipo_des=3 order by fecha_registro desc) * @calculo_salario;
set @isss = (select top(1) (porcentaje) from descuento_ley_detalle where id_tipo_des=1 order by fecha_registro desc) * @calculo_salario;
set @sueldo = @calculo_salario - (@afp + @isss) + @horas_extras;
```

Se usa la variable @calculo_salario para determinar los descuentos de ley de ISSS y AFP. Los porcentajes de estos descuentos son extraidos de una tabla que contiene información de estos, como el porcentaje, la categoria, fecha de registro, etc. El resultado de esto, se guarda en una variable llamada @sueldo

Cálculo dinámico del descuento de la renta en base al rango en el que se encuentra el sueldo

```
declare @ranin1 real = (select rango_inicial from rango_descuento where id_detalle_des=1)
declare @ranfin1 real = (select rango_final from rango_descuento where id_detalle_des=1)

if @sueldo > @ranin1 and @sueldo < @ranfin1
    begin
        declare @exceso real = 0
        declare @cuota real = 0
        set @descuento_renta = @sueldo;
    end

else if @sueldo > (select rango_inicial from rango_descuento where id_detalle_des=2) and @sueldo < (select rango_final from rango_descuento where id_detalle_des=2)
    begin
        declare @detalle int = (select top(1) id_detalle_des from descuento_ley_detalle where id_tipo_des = 2 and categoria = 2 order by fecha_registro desc)
        set @exceso = ((select rango_inicial-0.01 from rango_descuento where id_detalle_des=2))
        set @renta = (select top(1) porcentaje from descuento_ley_detalle where id_tipo_des = 2 and categoria = 2 order by fecha_registro desc)
        set @cuota = (select cuota from rango_descuento where id_detalle_des = @detalle)
    end

else if @sueldo > (select rango_inicial from rango_descuento where id_detalle_des=3) and @sueldo < (select rango_final from rango_descuento where id_detalle_des=3)
    begin
        set @exceso = (select rango_inicial-0.01 from rango_descuento where id_detalle_des=3)
        set @renta = (select top(1) porcentaje from descuento_ley_detalle where id_tipo_des = 2 and categoria = 3 order by fecha_registro desc)
        set @detalle = (select top(1) id_detalle_des from descuento_ley_detalle where id_tipo_des = 2 and categoria = 3 order by fecha_registro desc)
        set @cuota  = (select cuota from rango_descuento where id_detalle_des = @detalle)
    end

else if @sueldo > (select rango_inicial from rango_descuento where id_detalle_des=4)
    begin
        set @exceso = (select rango_inicial-0.01 from rango_descuento where id_detalle_des=4)
        set @renta = (select top(1) porcentaje from descuento_ley_detalle where id_tipo_des = 2 and categoria = 4 order by fecha_registro desc)
        set @detalle = (select top(1) id_detalle_des from descuento_ley_detalle where id_tipo_des = 2 and categoria = 4 order by fecha_registro desc)
        set @cuota  = (select cuota from rango_descuento where id_detalle_des = @detalle)
    end
```

Nuevamente usando la información que contienen las tablas que almacenan los detalles de descuentos de ley, se calcula el descuento de la renta en base a la categoría en la que entre el @sueldo de forma dinámica

Calculo del salario neto

Finalmente, se termina de calcular el descuento de la renta en base a el if anterior, y se determina el total a descontar al salario con todos los descuentos de ley y los laborales(si existieran), guardando esto en la variable @salario_neto

```
set @descuento_renta = ((@sueldo -  @exceso) * @renta) + @cuota;
set @descuentos_total = @afp + @isss + @descuento_renta;
set @salario_neto = @calculo_salario - @descuentos_total;
```

Creación de la boleta

Finalmente, se crea el encabezado y el detalle de la boleta con la información calculada previamente.

```
-- Creación del encabezado de la boleta
insert into boleta_pago_mensual(id_asistencia,id_mes,total_desc,salario_neto)
values(@asis,@pa_mes,round(@descuentos_total,2),round(@salario_neto,2));

declare @boleta int = (select id_boleta from boleta_pago_mensual where id_boleta = @@IDENTITY);

-- Creación de los detalles de la boleta
insert into
detalle_boleta (id_boleta,id_mov,concepto,monto)
values
(@boleta,2,'Salario devengado',round(@calculo_salario,2)),
(@boleta,3,'Descuento de AFP',round(@afp,2)),
(@boleta,3,'Descuento de ISSS',round(@isss,2)),
(@boleta,3,'Impuesto sobre la renta',round(@descuento_renta,2)),
(@boleta,1,'Horas/días no laborados',round(@descuento,2)),
(@boleta,2,'Horas extras',round(@horas_extras,2));
end
```

Procedure para insertar un nuevo empleado

```
go
create procedure pa_NuevoEmpleado
@nombres varchar(100),
@apellidos varchar(100),
@fecha_nacimiento date,
@correo_electronico varchar(60),
@dui varchar(12),
@nit varchar(20),
@telefono varchar(10),
@direccion varchar(250),
@id_departamento int ,
@id_sexo int
as
begin
insert into empleados(nombres,apellidos,fecha_nacimiento,correo_electronico,dui,nit,telefono,direccion,id_departamento,id_sexo) values
  (@nombres,@apellidos,@fecha_nacimiento,@correo_electronico,@dui,@nit,@telefono,@direccion,@id_departamento,@id_sexo)
end
```

Procedures que alimentan los DropDownList

```
go
create procedure pa_sexo
as
begin
select * from sexo;
end

go
create procedure pa_Departamento
as
begin
select * from departamento;
end

go
create procedure pa_SubDepartamento
@id_departamento int
as
begin
select * from subdepartamento where id_departamento=@id_departamento;
end

go
create procedure pa_Turno_Horario
as
begin
select * from turno_horario;
end

go
create procedure pa_Plaza
as
begin
select * from plaza;
end

go
create procedure pa_Mes
as
begin
select * from mes;
end
```

Procedure para insertar un contrato

```
go
create procedure pa_Contrato
@id_empleado int,
@id_plaza int ,
@id_turno int ,
@horas_semana int ,
@salario_base real
as
begin
insert into contrato(id_empleado,id_plaza,id_turno,horas_semana,salario_base)
values (@id_empleado,@id_plaza,@id_turno,@horas_semana,@salario_base);
end
```

Procedure para retornar a todos los empleados

```
go
create procedure [dbo].[pa_EmpleadosList]
as
begin
select CONCAT(nombres,apellidos)As'fullName',id_empleado from empleados;
end
GO
```

Procedure para retornar a todos aquellos empleados que no tengan un contrato

```
CREATE procedure [dbo].[pa_EmpSinContrato]
as
begin
select CONCAT(nombres,apellidos)As'fullName',e.id_empleado As 'id_empleado'
from empleados e left join contrato c on e.id_empleado=c.id_empleado where c.id_empleado is null;
end
```

Vista para juntar la mayor cantidad de tablas.

```
create View Reporte_Boleta_Pago
as
select e.id_empleado,boleta.id_boleta,CONCAT(e.nombres,' ',e.apellidos)as
  nombre,p.plaza,c.salario_base,a.dias_asistidos,a.total_horas_realizadas,boleta.id_mes as mes,YEAR(boleta.fecha_generacion)As
  año,tipo.movimiento,detalle.concepto,detalle.monto,boleta.salario_neto from empleados as e inner join contrato as c on e.id_empleado=c.id_empleado
  inner join plaza as p on c.id_plaza=p.id_plaza inner join asistencia_mensual as a on e.id_empleado=a.id_empleado inner join boleta_pago_mensual as
  boleta on a.id_asistencia=boleta.id_asistencia inner join detalle_boleta as detalle on boleta.id_boleta=detalle.id_boleta inner join tipo_movimiento as
   tipo on detalle.id_mov=tipo.id_movimiento
```

Procedure para buscar la boleta de una persona, con su respectivo año y mes.

```
create procedure GenerarBoleta
@id_empleado int,
@año int,
@mes int
as
begin
select * from Reporte_Boleta_Pago where id_empleado=@id_empleado AND año=@año AND mes=@mes;
end
```

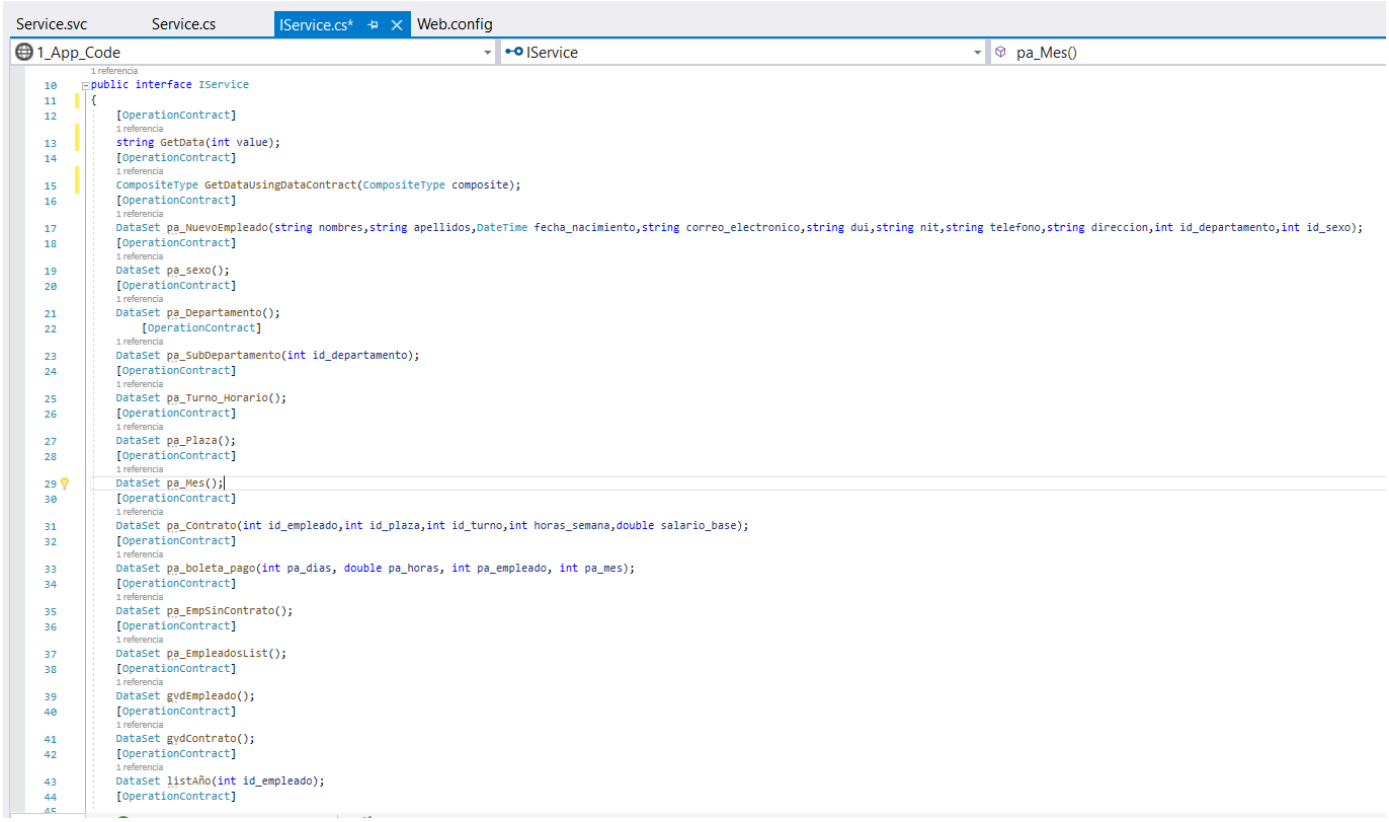
Procedure para llenar los GridView.

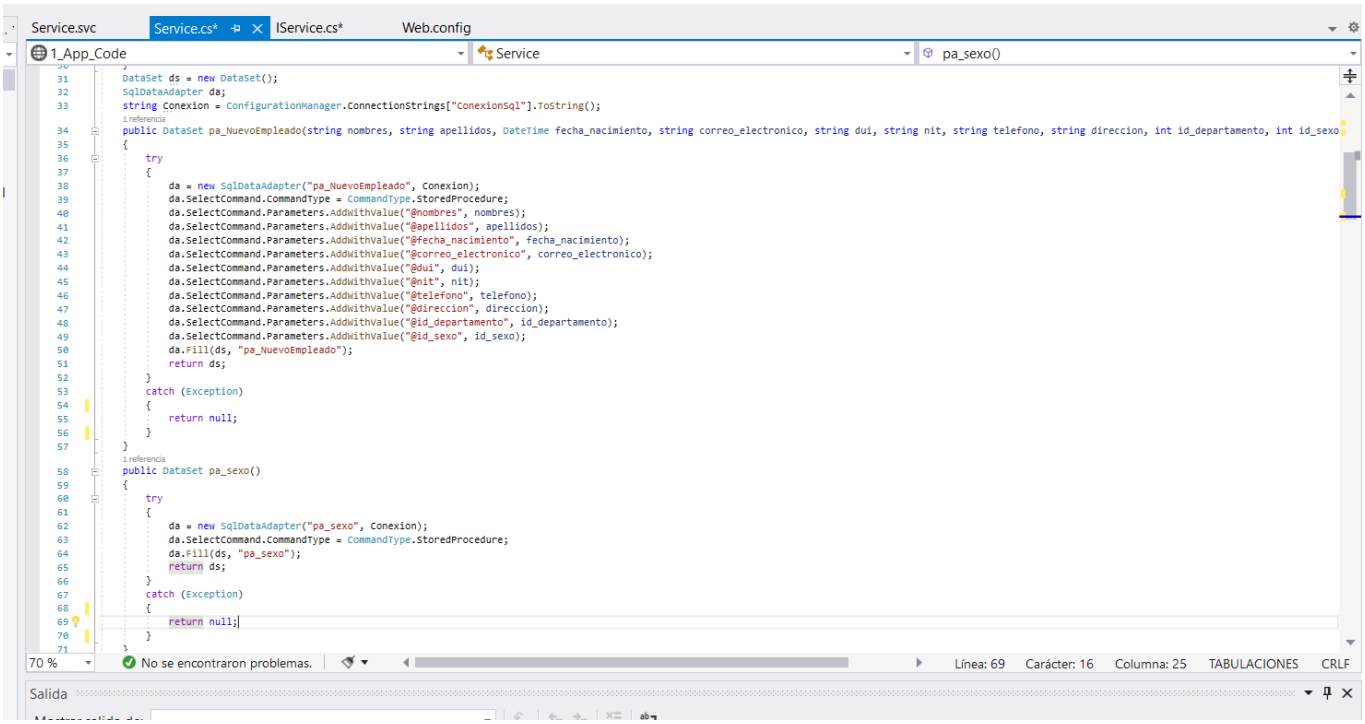
```
create procedure gvdEmpleado
as
begin
select
  e.id_empleado,e.nombres,e.apellidos,se.sexo,e.fecha_nacimiento,e.correo_electronico,e.dui,e.nit,e.fecha_registro,e.telefono,e.direccion,d.nombre_depart
  amento,s.nombre_subdepartamento from empleados e inner join subdepartamento s on e.id_departamento=s.id_subdepartamento inner join departamento d on
  s.id_departamento=d.id_departamento inner join sexo se on e.id_sexo=se.id_sexo;
end
go
create procedure gvdContrato
as
begin
select c.id_contrato,concat(e.nombres,' ', e.apellidos)as 'nombre',p.plaza,t.turno,horas_semana,salario_base,c.fecha_contrato,p.plaza,t.turno from
  contrato c inner join empleados e on c.id_empleado=e.id_empleado inner join plaza p on c.id_plaza=p.id_plaza inner join turno_horario t on
  c.id_turno=t.id_turno;
end
```

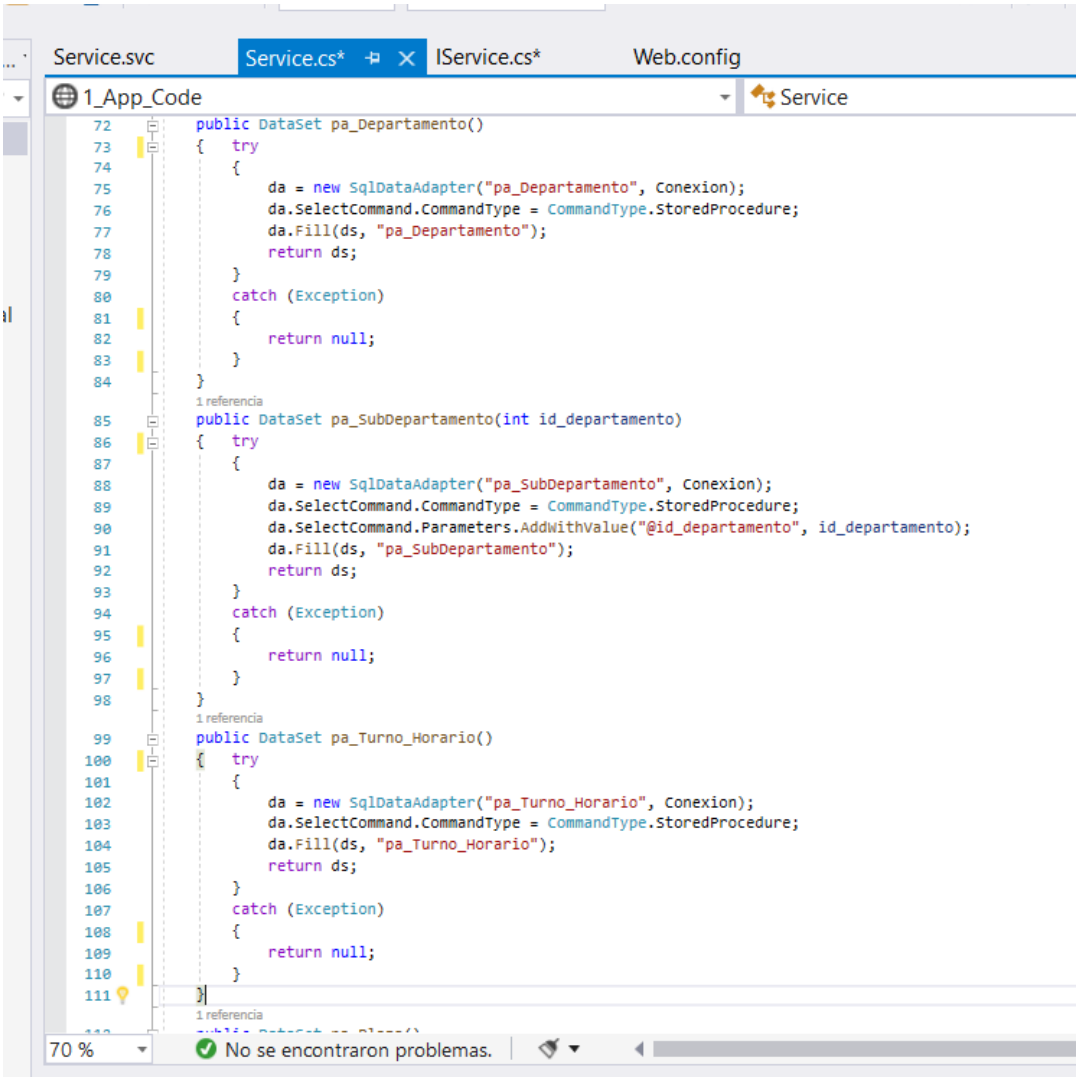
SEGUNDA PARTE (Prueba de ejecución - Habilidades) …… Ponderación 50%.

<u>Indicaciones</u> – Crear un servicio web para realizar la búsqueda:

1. Crear un servicio web que usando los procedimientos almacenados creados en la primera parte pueda agregar empleados, salarios, días trabajados, horas extras y los descuentos de ley requeridos para generar la información del siguiente numeral.




Service.svc
Service.cs*
IService.cs*
Web.config
1_App_Code
Service
pa_sexo()
DataSet ds = new DataSet();
SqlDataAdapter da;
string Conexion = ConfigurationManager.ConnectionStrings["ConexionSql"].ToString();
1 referencia
public DataSet pa_NuevoEmpleado(string nombres, string apellidos, DateTime fecha_nacimiento, string correo_electronico, string dui, string nit, string telefono, string direccion, int id_departamento, int id_sexo
{
try
{
da = new SqlDataAdapter("pa_NuevoEmpleado", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.SelectCommand.Parameters.AddWithValue("@nombres", nombres);
da.SelectCommand.Parameters.AddWithValue("@apellidos", apellidos);
da.SelectCommand.Parameters.AddWithValue("@fecha_nacimiento", fecha_nacimiento);
da.SelectCommand.Parameters.AddWithValue("@correo_electronico", correo_electronico);
da.SelectCommand.Parameters.AddWithValue("@dui", dui);
da.SelectCommand.Parameters.AddWithValue("@nit", nit);
da.SelectCommand.Parameters.AddWithValue("@telefono", telefono);
da.SelectCommand.Parameters.AddWithValue("@direccion", direccion);
da.SelectCommand.Parameters.AddWithValue("@id_departamento", id_departamento);
da.SelectCommand.Parameters.AddWithValue("@id_sexo", id_sexo);
da.Fill(ds, "pa_NuevoEmpleado");
return ds;
}
catch (Exception)
{
return null;
}
}
1 referencia
public DataSet pa_sexo()
{
try
{
da = new SqlDataAdapter("pa_sexo", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "pa_sexo");
return ds;
}
catch (Exception)
{
return null;
}
}
70 %
No se encontraron problemas.
Línea: 69
Carácter: 16
Columna: 25
TABULACIONES
CRLF
Salida

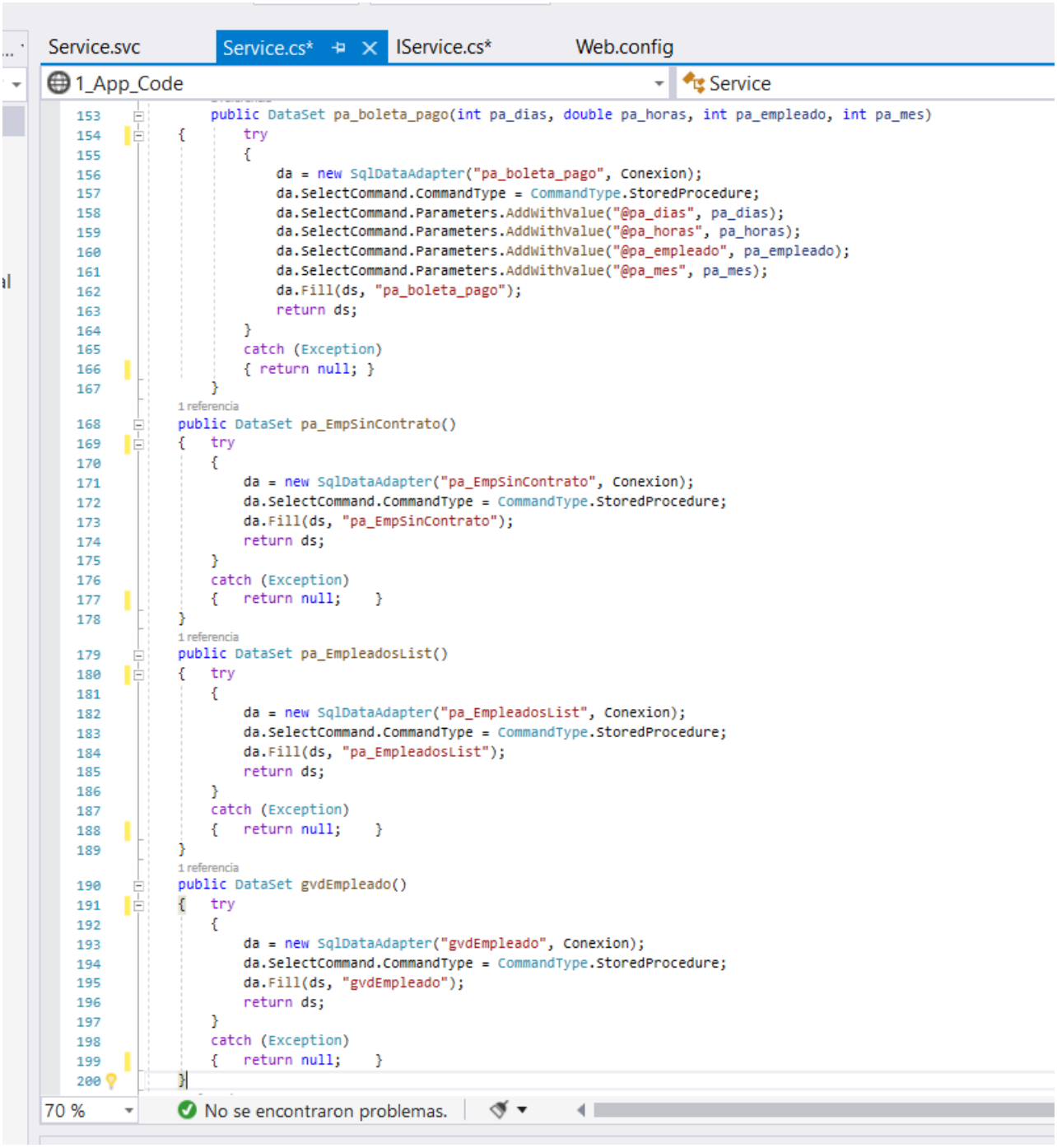

Service.svc
Service.cs*
IService.cs*
Web.config
1_App_Code
Service
public DataSet pa_Departamento()
{ try
{
da = new SqlDataAdapter("pa_Departamento", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "pa_Departamento");
return ds;
}
catch (Exception)
{
return null;
}
}
1 referencia
public DataSet pa_SubDepartamento(int id_departamento)
{ try
{
da = new SqlDataAdapter("pa_SubDepartamento", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.SelectCommand.Parameters.AddWithValue("@id_departamento", id_departamento);
da.Fill(ds, "pa_SubDepartamento");
return ds;
}
catch (Exception)
{
return null;
}
}
1 referencia
public DataSet pa_Turno_Horario()
{ try
{
da = new SqlDataAdapter("pa_Turno_Horario", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "pa_Turno_Horario");
return ds;
}
catch (Exception)
{
return null;
}
}
1 referencia
70 %
No se encontraron problemas.

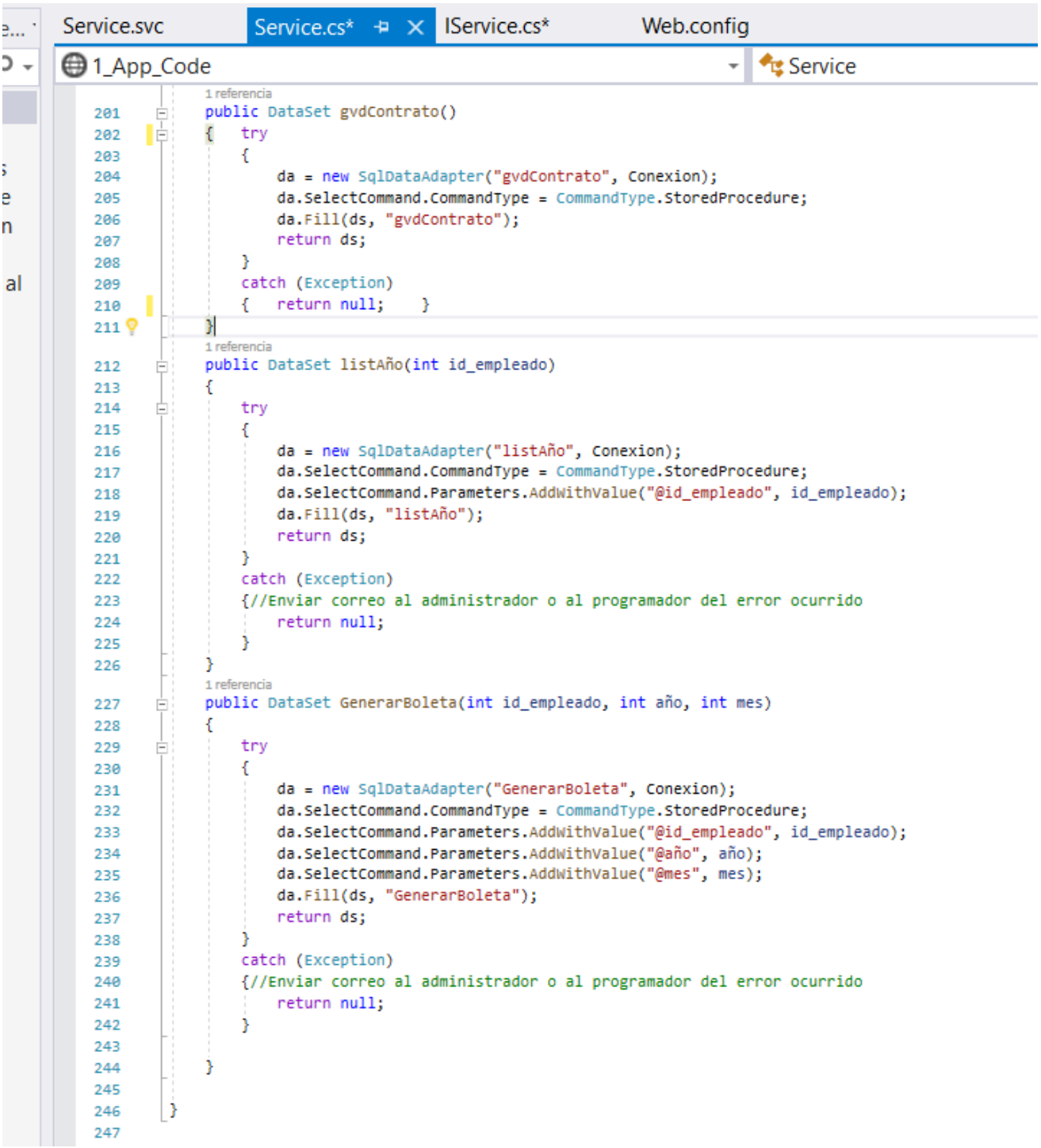
Service.svc | Service.cs* | IService.cs* | Web.config

1_App_Code | Service

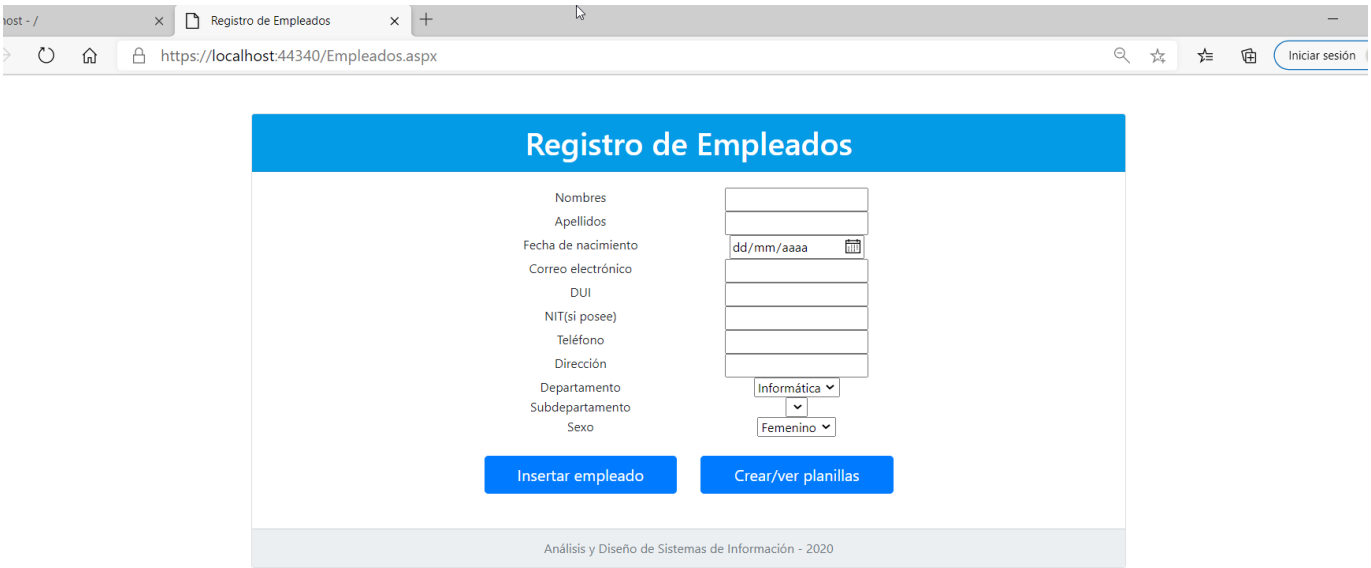
```
112     public DataSet pa_Plaza()
113     {   try
114         {
115             da = new SqlDataAdapter("pa_Plaza", Conexion);
116             da.SelectCommand.CommandType = CommandType.StoredProcedure;
117             da.Fill(ds, "pa_Plaza");
118             return ds;
119         }
120         catch (Exception)
121         {   return null;
122         }
123     }
        1 referencia
124     public DataSet pa_Mes()
125     {   try
126         {
127             da = new SqlDataAdapter("pa_Mes", Conexion);
128             da.SelectCommand.CommandType = CommandType.StoredProcedure;
129             da.Fill(ds, "pa_Mes");
130             return ds;
131         }
132         catch (Exception)
133         {   return null;
134         }
135     }
        1 referencia
136     public DataSet pa_Contrato(int id_empleado, int id_plaza, int id_turno, int horas_semana, double salario_base)
137     {   try
138         {
139             da = new SqlDataAdapter("pa_Contrato", Conexion);
140             da.SelectCommand.CommandType = CommandType.StoredProcedure;
141             da.SelectCommand.Parameters.AddWithValue("@id_empleado", id_empleado);
142             da.SelectCommand.Parameters.AddWithValue("@id_plaza", id_plaza);
143             da.SelectCommand.Parameters.AddWithValue("@id_turno", id_turno);
144             da.SelectCommand.Parameters.AddWithValue("@horas_semana", horas_semana);
145             da.SelectCommand.Parameters.AddWithValue("@salario_base", salario_base);
146             da.Fill(ds, "pa_Contrato");
147             return ds;
148         }
149         catch (Exception)
150         {   return null;
151         }
152     }
```

70 % | No se encontraron problemas.


Service.svc
Service.cs*
IService.cs*
Web.config
1_App_Code
Service
public DataSet pa_boleta_pago(int pa_dias, double pa_horas, int pa_empleado, int pa_mes)
{
try
{
da = new SqlDataAdapter("pa_boleta_pago", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.SelectCommand.Parameters.AddWithValue("@pa_dias", pa_dias);
da.SelectCommand.Parameters.AddWithValue("@pa_horas", pa_horas);
da.SelectCommand.Parameters.AddWithValue("@pa_empleado", pa_empleado);
da.SelectCommand.Parameters.AddWithValue("@pa_mes", pa_mes);
da.Fill(ds, "pa_boleta_pago");
return ds;
}
catch (Exception)
{ return null; }
}
1 referencia
public DataSet pa_EmpSinContrato()
{ try
{
da = new SqlDataAdapter("pa_EmpSinContrato", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "pa_EmpSinContrato");
return ds;
}
catch (Exception)
{ return null; }
}
1 referencia
public DataSet pa_EmpleadosList()
{ try
{
da = new SqlDataAdapter("pa_EmpleadosList", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "pa_EmpleadosList");
return ds;
}
catch (Exception)
{ return null; }
}
1 referencia
public DataSet gvdEmpleado()
{ try
{
da = new SqlDataAdapter("gvdEmpleado", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "gvdEmpleado");
return ds;
}
catch (Exception)
{ return null; }
}
70 %
No se encontraron problemas.


Service.svc
Service.cs*
IService.cs*
Web.config
1_App_Code
Service
1 referencia
public DataSet gvdContrato()
{ try
{
da = new SqlDataAdapter("gvdContrato", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.Fill(ds, "gvdContrato");
return ds;
}
catch (Exception)
{ return null; }
}
1 referencia
public DataSet listAño(int id_empleado)
{
try
{
da = new SqlDataAdapter("listAño", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.SelectCommand.Parameters.AddWithValue("@id_empleado", id_empleado);
da.Fill(ds, "listAño");
return ds;
}
catch (Exception)
{//Enviar correo al administrador o al programador del error ocurrido
return null;
}
}
1 referencia
public DataSet GenerarBoleta(int id_empleado, int año, int mes)
{
try
{
da = new SqlDataAdapter("GenerarBoleta", Conexion);
da.SelectCommand.CommandType = CommandType.StoredProcedure;
da.SelectCommand.Parameters.AddWithValue("@id_empleado", id_empleado);
da.SelectCommand.Parameters.AddWithValue("@año", año);
da.SelectCommand.Parameters.AddWithValue("@mes", mes);
da.Fill(ds, "GenerarBoleta");
return ds;
}
catch (Exception)
{//Enviar correo al administrador o al programador del error ocurrido
return null;
}
}

2. Crear una página web para agregar empleados (considere todas las validaciones de datos) y otro WebForm para el proceso de generar la planilla de pagos la cual se debe mostrar en un GridView (No es necesario imprimirla)



id_empleado	nombres	apellidos	sexo	fecha_nacimiento	correo_electronico	dui	nit	fecha_registro	telefono	direccion	nombre_departamento	nombre_subdepartamento
1	Pedro	Molina	Masculino	10/10/1990 00:00:00	pedro@gmail.com	0615487-0		20/10/2020 00:00:00	7845-7845	San Salvador	Informática	Desarrollo
2	Carlos	Rodriguez	Masculino	10/10/1990 00:00:00	carlos@gmail.com	0784510-0		20/10/2020 00:00:00	7845-7845	San Salvador	Informática	Desarrollo
3	fatima	miranda	Femenino	14/7/2000 00:00:00	prueba@mail.com	12345678-9	9999-999999-999-9	20/10/2020 00:00:00	(999) 9999	San Salavador	Informática	Desarrollo

Validaciones del formulario de empleados

Empleados.aspx.cs

Archivos varios

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Text.RegularExpressions;

namespace WFParcial3ADS
{
    public partial class Empleados : System.Web.UI.Page
    {
        ServiceReference1.ServiceClient wcf = new ServiceReference1.ServiceClient();
        DataSet dsDpts = new DataSet();

        Regex valNombre = new Regex("^[A-Za-záéíóúÁÉÍÓÚñÑ ]{0,100}$");
        Regex valApellido = new Regex("^[A-Za-záéíóúÁÉÍÓÚñÑ ]{0,100}$");
        Regex valEmail = new Regex(@"^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$");
        Regex valDui = new Regex(@"^\d{8}\-\d$");
        Regex valNit = new Regex(@"^\d{4}\-\d{6}\-\d{3}\-\d{1}$");
        Regex valTelefono = new Regex(@"^\d{4}\-\d{4}$");
        Regex valDireccion = new Regex(@"^[A-Za-záéíóúÁÉÍÓÚñÑ #,1-9]{0,250}$");

        DataSet DsEmpleados = new DataSet();
        protected void refreshGvd()
        {
            DsEmpleados = wcf.gvdEmpleado();
            gvdEmpleados.DataSource = DsEmpleados;
            gvdEmpleados.DataBind();
        }
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack)
            {
                dsDpts = wcf.pa_Departamento();
                ddlDepartamento.DataSource = dsDpts;
                ddlDepartamento.DataMember = "pa_Departamento";
                ddlDepartamento.DataTextField = "nombre_departamento";
                ddlDepartamento.DataValueField = "id_departamento";
                ddlDepartamento.DataBind();
            }
            if (!Page.IsPostBack)
            {
                dsDpts = wcf.pa_sexo();
                ddlSexo.DataSource = dsDpts;
                ddlSexo.DataMember = "pa_sexo";
                ddlSexo.DataTextField = "sexo";
                ddlSexo.DataValueField = "id_sexo";
                ddlSexo.DataBind();
            }
            if (!Page.IsPostBack)
            {
                refreshGvd();
            }
        }
```

70 % No se encontraron problemas.

Empleados.aspx.cs

Archivos varios | WFParcial3ADS.Empleados | Page_Load(object sender, EventArgs e)

```
protected void ddlDepartamento_SelectedIndexChanged(object sender, EventArgs e)
{
    dsDpts = wcf.pa_SubDepartamento(int.Parse(ddlDepartamento.SelectedValue));
    ddlSubdepartamento.DataSource = dsDpts;
    ddlSubdepartamento.DataMember = "pa_SubDepartamento";
    ddlSubdepartamento.DataTextField = "nombre_subdepartamento";
    ddlSubdepartamento.DataValueField = "id_subdepartamento";
    ddlSubdepartamento.DataBind();
}

protected void btnInsertarEmpleado_Click(object sender, EventArgs e)
{
    try
    {
        //Validación de entrada de datos usando las expresiones regulares
        if (!valNombre.IsMatch(txtNombre.Text))
        {
            lblError.Text = "*Error. Sólo se permiten letras (Máx. 100)";
            return;
        }
        if (!valApellido.IsMatch(txtApellido.Text))
        {
            lblError.Text = "*Error. Sólo se permiten letras (Máx. 100)";
            return;
        }
        if (!valEmail.IsMatch(txtCorreo.Text))
        {
            lblError.Text = "*Error, el formato de correo no es válido. (Máx. 50)";
            return;
        }
        if (!valDui.IsMatch(txtDui.Text))
        {
            lblError.Text = "*Error, el formato de correo no es válido. (Máx. 50)";
            return;
        }
        if (!valNit.IsMatch(txtNit.Text))
        {
            lblError.Text = "*Error, el formato de correo no es válido. (Máx. 50)";
            return;
        }
        if (!valTelefono.IsMatch(txtTelefono.Text))
        {
            lblError.Text = "*Error, el formato de correo no es válido. (Máx. 50)";
            return;
        }
        if (!valDireccion.IsMatch(txtDireccion.Text))
        {
            lblError.Text = "*Error, el formato de correo no es válido. (Máx. 50)";
            return;
        }

        dsDpts = wcf.pa_NuevoEmpleado(txtNombre.Text, txtApellido.Text,DateTime.Parse(datFecha.Text), txtCorreo.Text,txtDui.Text,txtNit.Text,txtTelefono.Text,txtDireccion.Text, int.Parse(ddlSubdepartamento.SelectedValue), int.Parse(ddlSexo.SelectedValue));
        refreshGvd();
        if (dsDpts != null)
        {
```

70 % | No se encontraron problemas. | Línea: 35 | Carácte

Registro de Contratos

Empleado
Plaza: Programador
Turno: Matutino
Horas a la semana
Salario base

Crear contrato

id_contrato	nombre	plaza	turno	horas_semana	salario_base	fecha_contrato	plaza1	turno1
1	Pedro Molina	Programador	Matutino	40	500	20/10/2020 00:00:00	Programador	Matutino
2	Carlos Rodriguez	Programador	Vespertino	35	500	20/10/2020 00:00:00	Programador	Vespertino
3	fatima miranda	consultor	Matutino	44	800	20/10/2020 00:00:00	consultor	Matutino

Análisis y Diseño de Sistemas de Información - 2020

Registro de Asistencia mensual

Empleado: PedroMolina
Mes: Enero
Dias asistidos
Total horas realizadas

Insertar asistencia y crear planilla

Empleado: PedroMolina
Año: 2020
Mes: Enero

Buscar planilla

Análisis y Diseño de Sistemas de Información - 2020

id_empleado	id_boleta	nombre	plaza	salario_base	dias_asistidos	total_horas_realizadas	mes	año	movimiento	concepto	monto	salario_neto
1	1	Pedro Molina	Programador	500	28	150	1	2020	Abono	Salario devengado	437.5	392.66
1	1	Pedro Molina	Programador	500	28	150	1	2020	Descuento de ley	Descuento de AFP	31.72	392.66
1	1	Pedro Molina	Programador	500	28	150	1	2020	Descuento de ley	Descuento de ISSS	13.13	392.66
1	1	Pedro Molina	Programador	500	28	150	1	2020	Descuento de ley	Impuesto sobre la renta	0	392.66
1	1	Pedro Molina	Programador	500	28	150	1	2020	Descuento	Horas/días no laborados	31.25	392.66
1	1	Pedro Molina	Programador	500	28	150	1	2020	Abono	Horas extras	0	392.66

TERCERA PARTE (Escala de autoevaluación de valores) ……. Ponderación 10%.

Indicaciones: Rúbrica para autoevaluación: **Integridad**
Mediante esta rúbrica se autoevaluará las actitudes y los valores mostrados durante sus clases, deberá asignarse una nota entre 1 a 10, según lo planteado, haciendo una marca en la casilla correspondiente.

Actitudes mostradas	1	2	3	4	5	6	7	8	9	10
1. Trabajo con tolerancia y buen agrado con los miembros de mi equipo.										x
2. Me intereso por aprender de los demás compañeros inscritos en la materia y también de los miembros de mi equipo de trabajo.										x
3. Realizo aportaciones académicas con el objetivo de mejorar las tareas extra aulas asignadas.										x
4. Cumplo con la entrega a tiempo de las obligaciones definidas en la materia y al equipo de trabajo.										x
5. Soy íntegro en mis compromisos académicos de la materia y con los integrantes de mi equipo de trabajo.										x
Sumatoria										50
Promedio (total/5)										10