PROYECTO NO.1

HUGO EDUARDO JAUREGUI JEREZ

Mini proyecto que da toda la documentación de un software de pago de planilla de una joyería.

PROYECTO NO.1

ENTREVISTA A LA DIRECTORA DE RECURSOS HUMANOS.

1. ¿Qué sistema tiene aquí para la administración de la nómina?
   R: // Se utiliza el sistema IDEA.Systems para el control y el manejo de la nómina.
2. ¿Cuáles serían las principales formas de pago de nómina?
   R: // se tienen dos formas
   1) La forma manual en la que se emite un cheque al empleado una vez que haya sido determinada la nómina en el sistema IDEA.
   2) Se captura en la web del Banco los abonos por empleado de acuerdo al reporte de pago emitido en el sistema IDEA
3. ¿Tienen actualmente un ciclo de mantenimiento a los sistemas de nómina?
   R: // No, la última versión que tienen data de Septiembre del 2016.
4. ¿Cómo se arreglan los problemas técnicos derivados de la falta de mantenimiento a los sistemas de nómina?
   R: // A través de operaciones manuales.
5. ¿Cuáles son las principales ventajas del sistema actual?
   R: // Que puede manipularse manualmente cualquier error.
6. ¿Cuáles son las principales desventajas del actual sistema de nómina?
   R: // Falta de automatización y enlace entre sedes y sistema contable, actualmente se posee un 60% de utilización de Excel para el cumplimiento de tareas.

DATOS A TOMAR EN CUENTA:

1. Cada usuario tiene un password propio por lo consiguiente debe haber una tabla llamada trabajadores o algo así.
2. Actualmente todo el proceso de cálculo es llevado en un Excel y solo se registra en el sistema Ideasystems.
3. Este sistema no tiene medio de comunicación con otras sucursales por lo consiguiente pensé en levantar una base de datos para que todos puedan almacenar sus datos en una sola tabla.
4. Dependen demasiado de Excel y del área de contabilidad para realizar los cálculos. Por lo cual podremos tener error humano.

DIAGRAMA DE FLUJO.

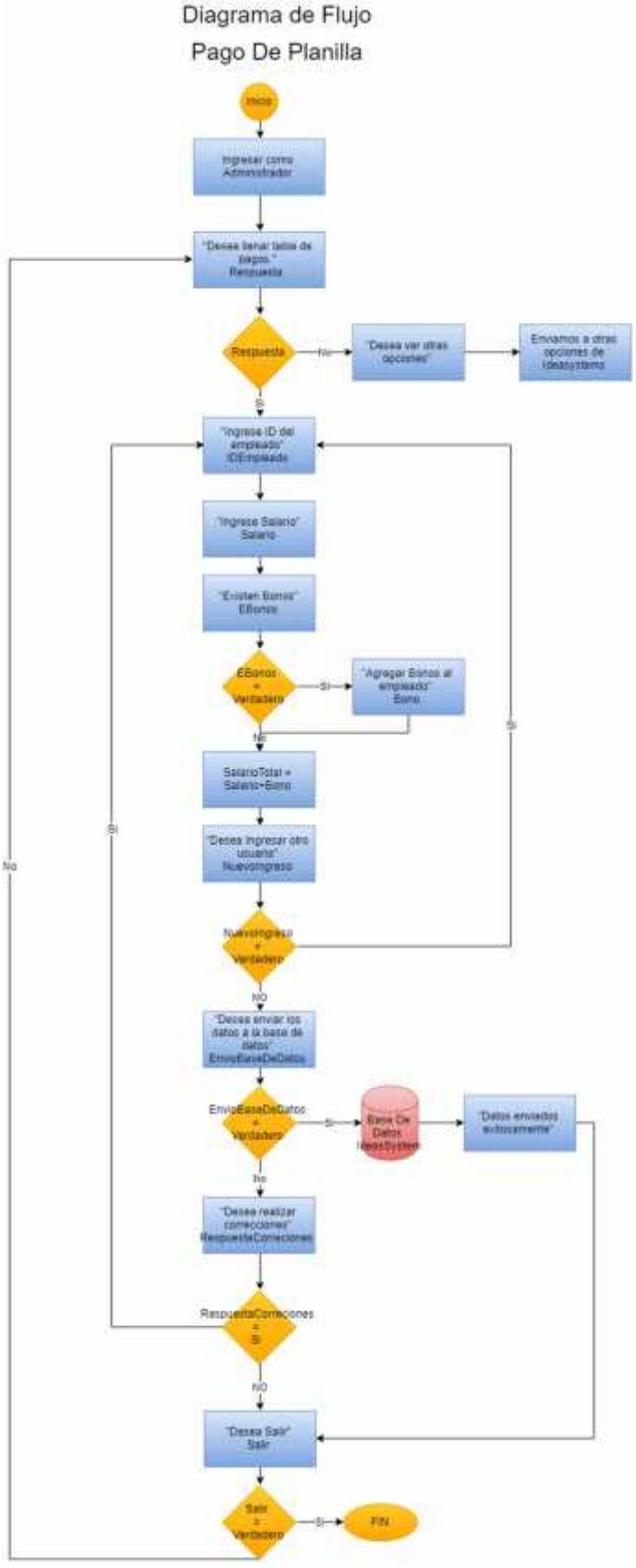


DIAGRAMA CASOS DE USO:

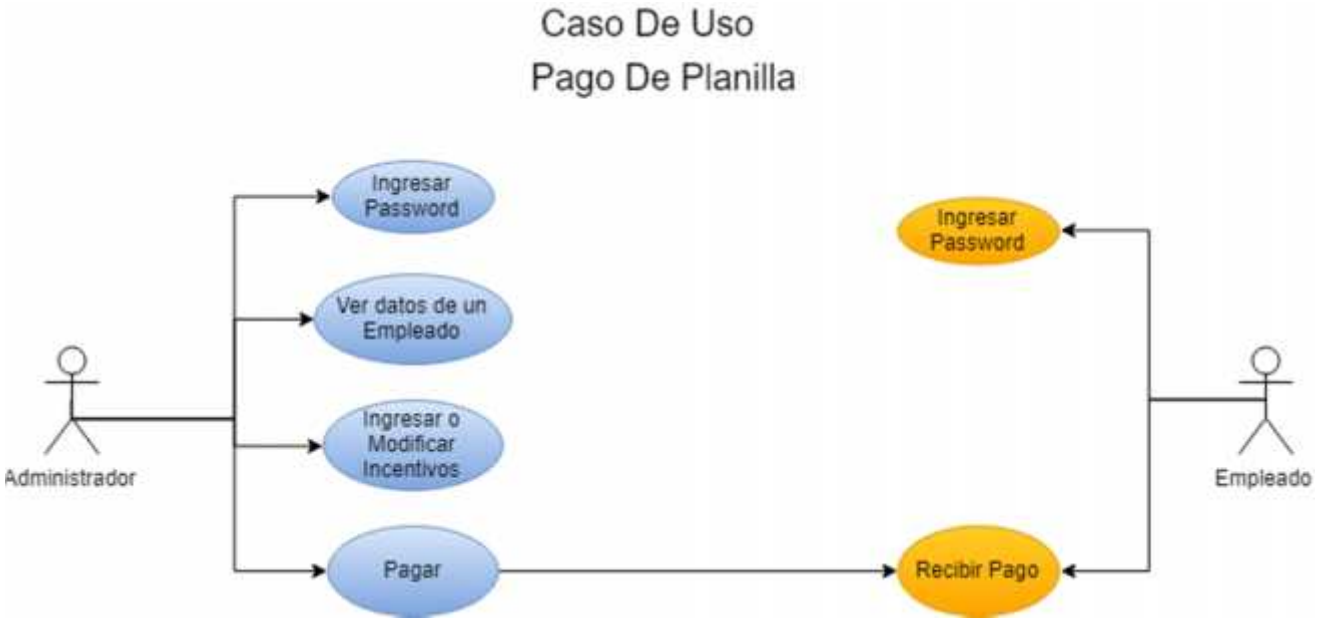


DIAGRAMA ENTIDAD RELACION:

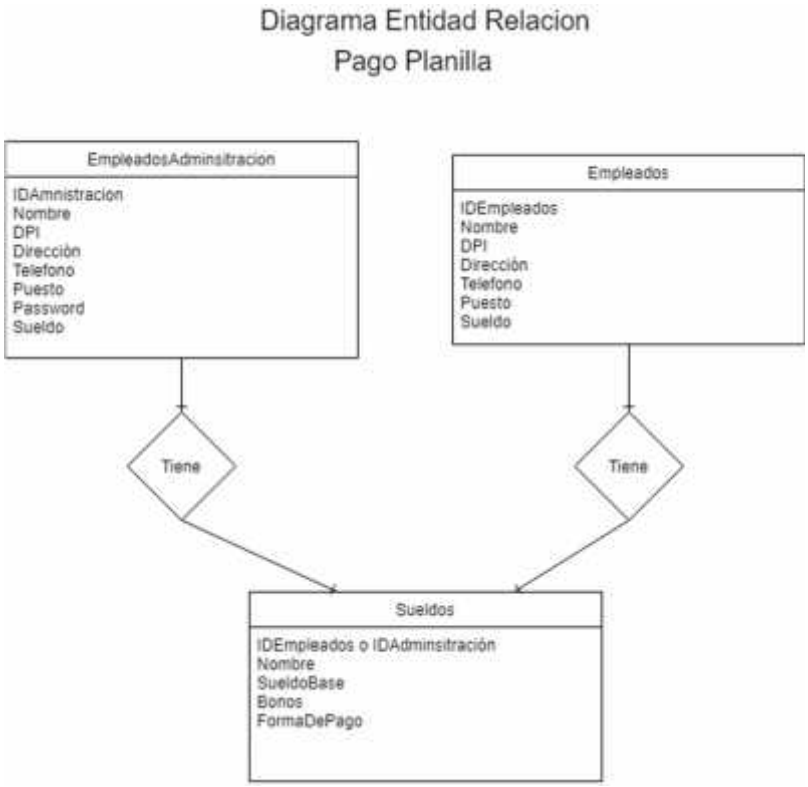


DIAGRAMA PAD:

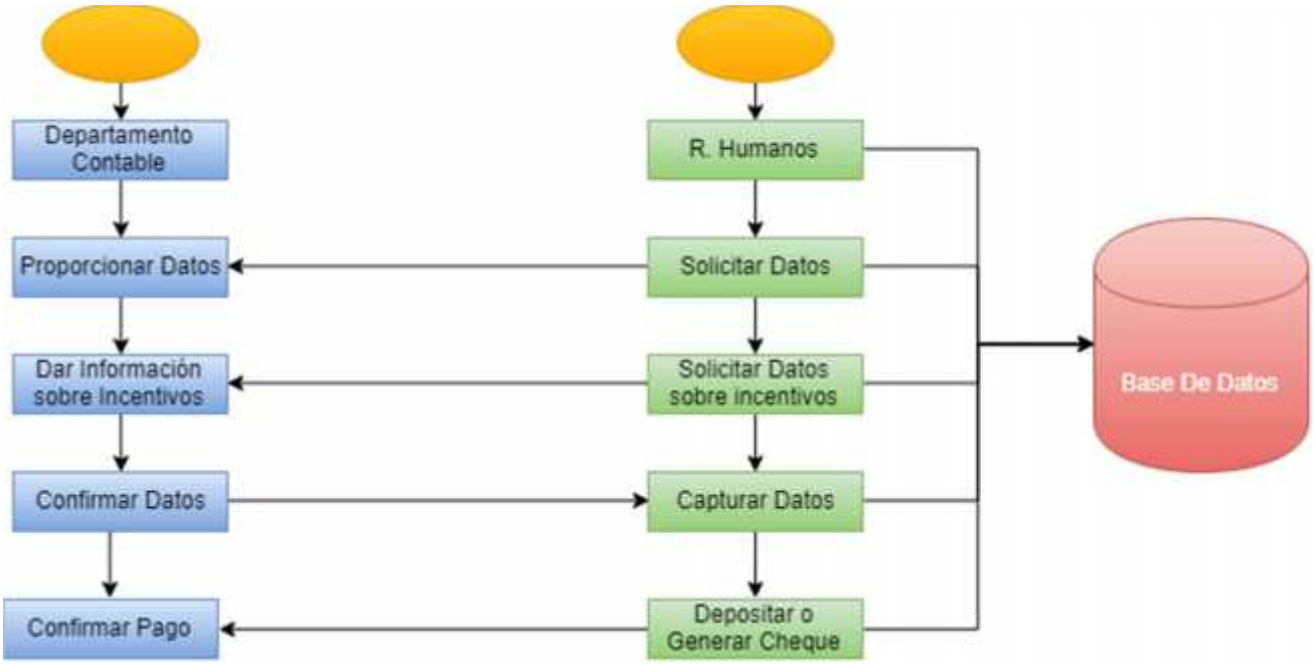
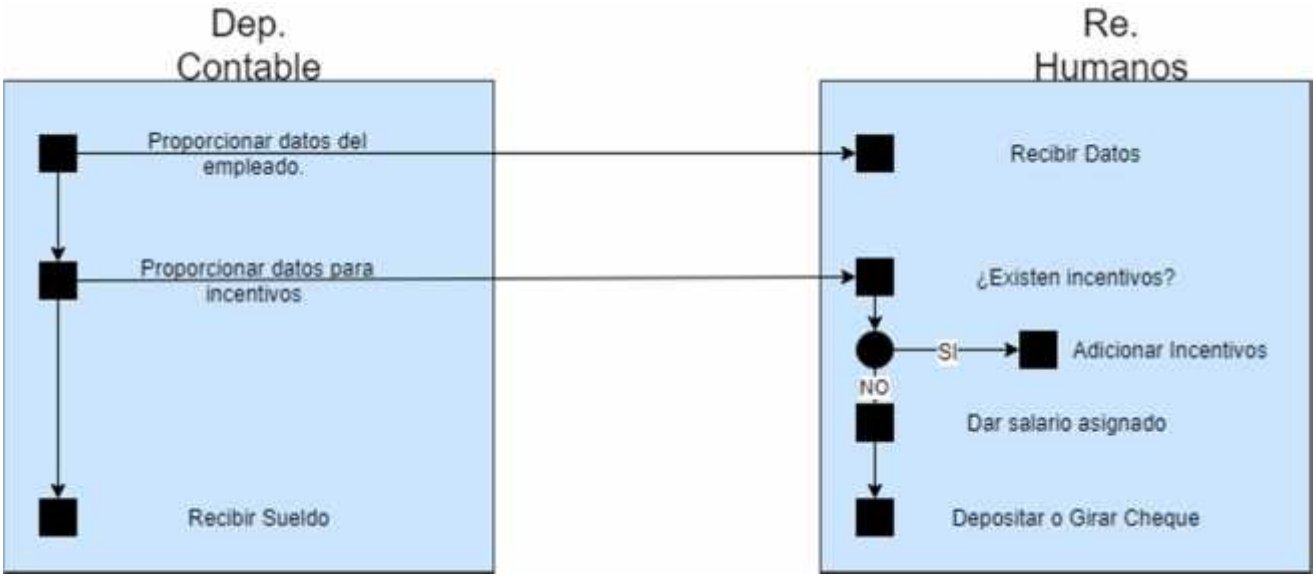


DIAGRAMA RAD:



ANEXOS:

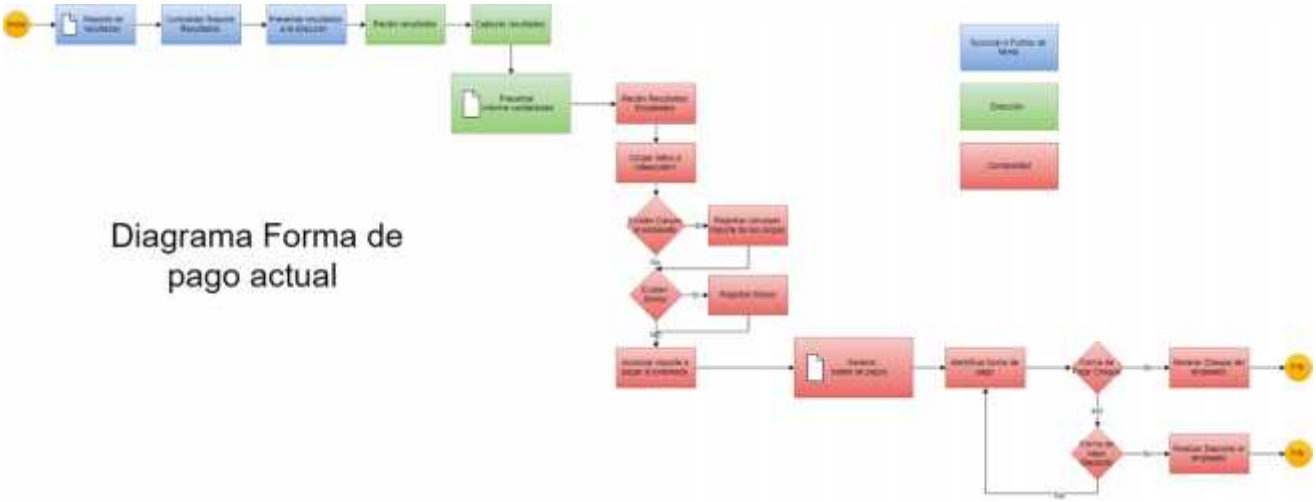
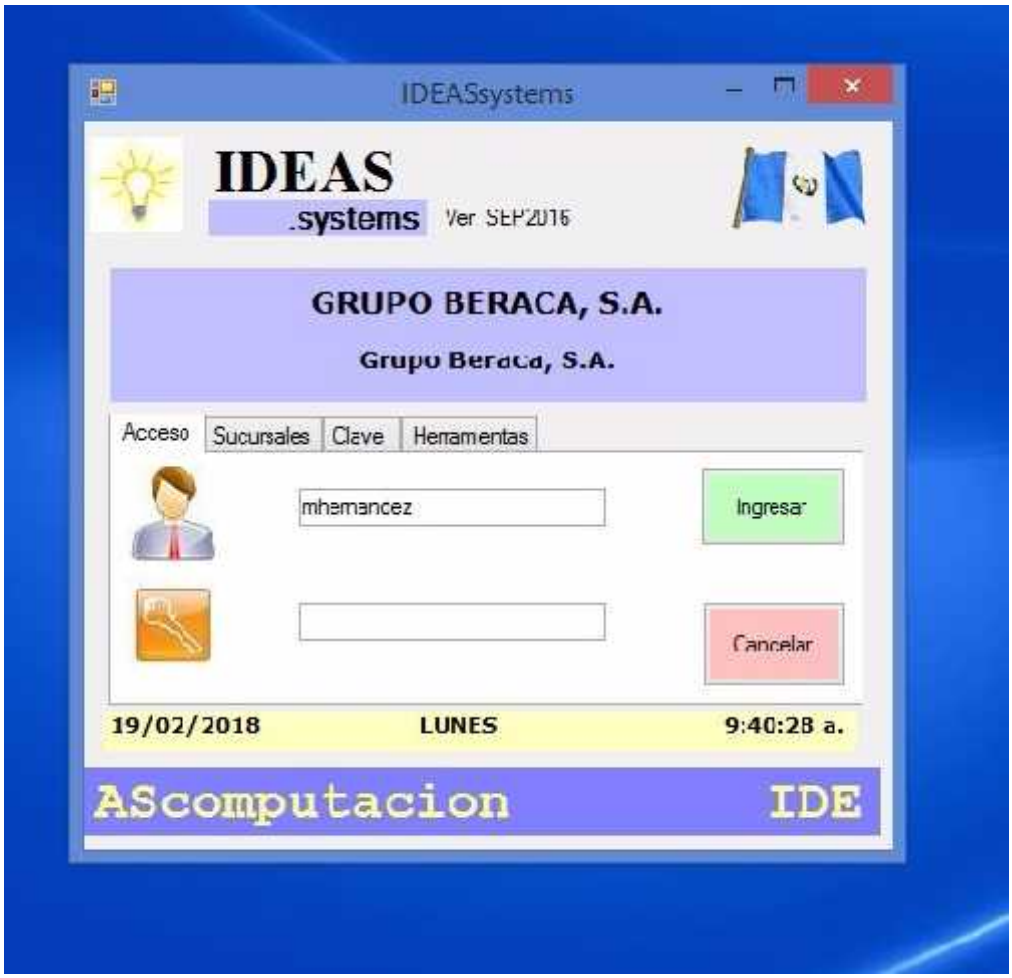
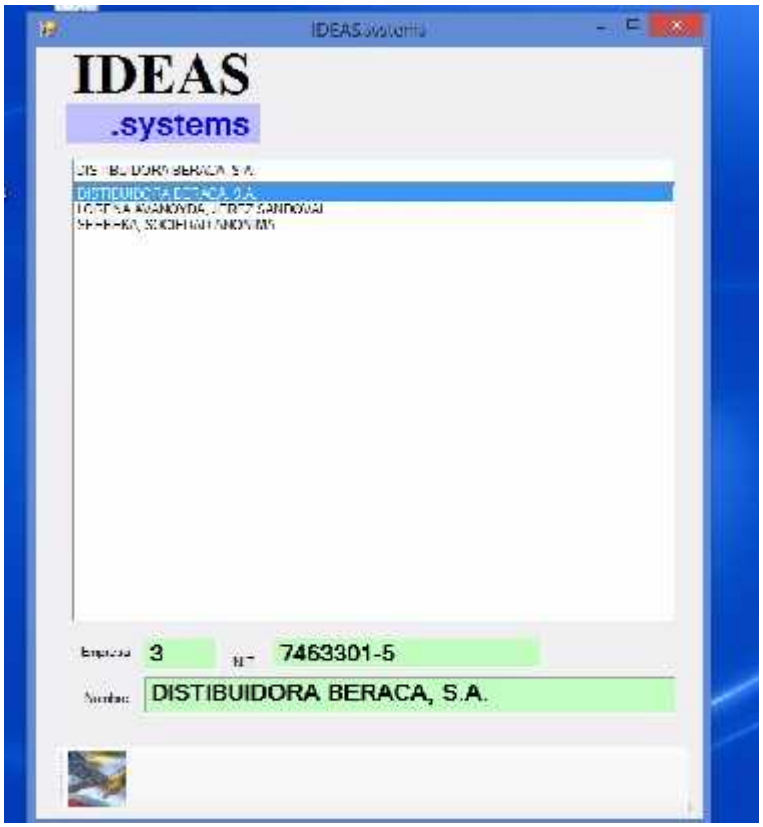
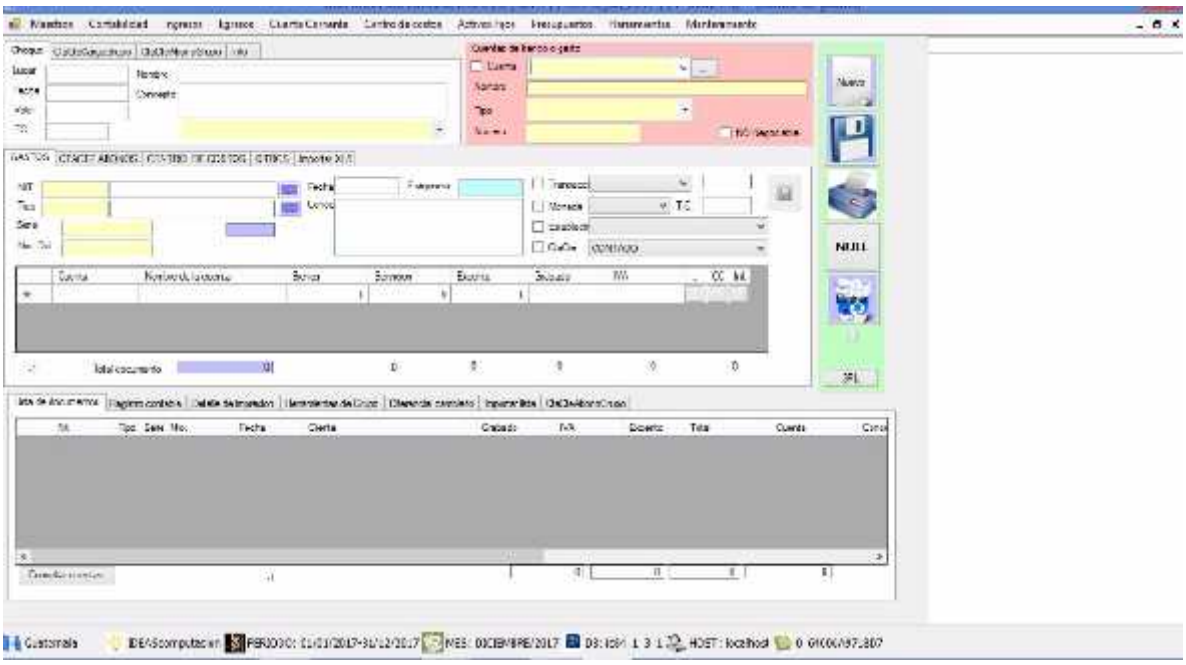


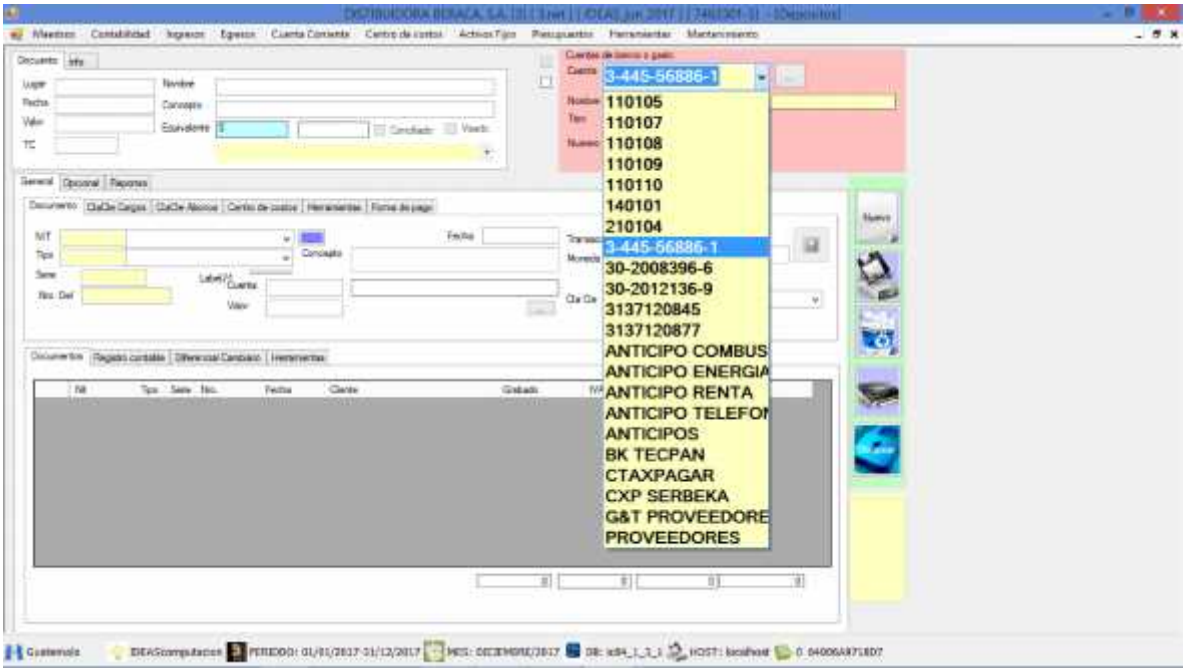
Diagrama Forma de pago actual

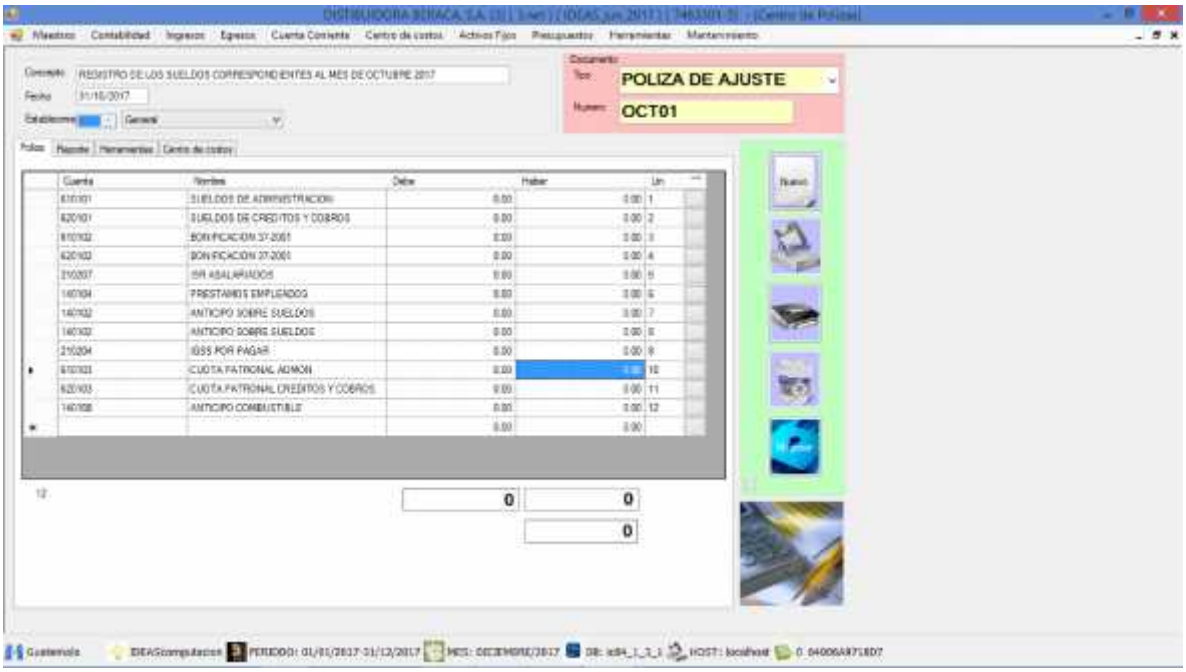
CAPTURAS IDEASYSTEMS:




IDEAS
.systems
DISTIBUIDORA BERACA, S.A.
Empresa 3
N- 7463301-5
Nombre DISTIBUIDORA BERACA, S.A.




3-445-56886-1
110105
110107
110108
110109
110110
140101
210104
3-445-56886-1
30-2008396-6
30-2012136-9
3137120845
3137120877
ANTICIPO COMBUS
ANTICIPO ENERGIA
ANTICIPO RENTA
ANTICIPO TELEFO
ANTICIPOS
BK TECPAN
CTAXPAGAR
CXP SERBEKA
G&T PROVEEDORE
PROVEEDORES


POLIZA DE AJUSTE
OCT01

SISTEMAS DE CONTROL:

DEFINICION:

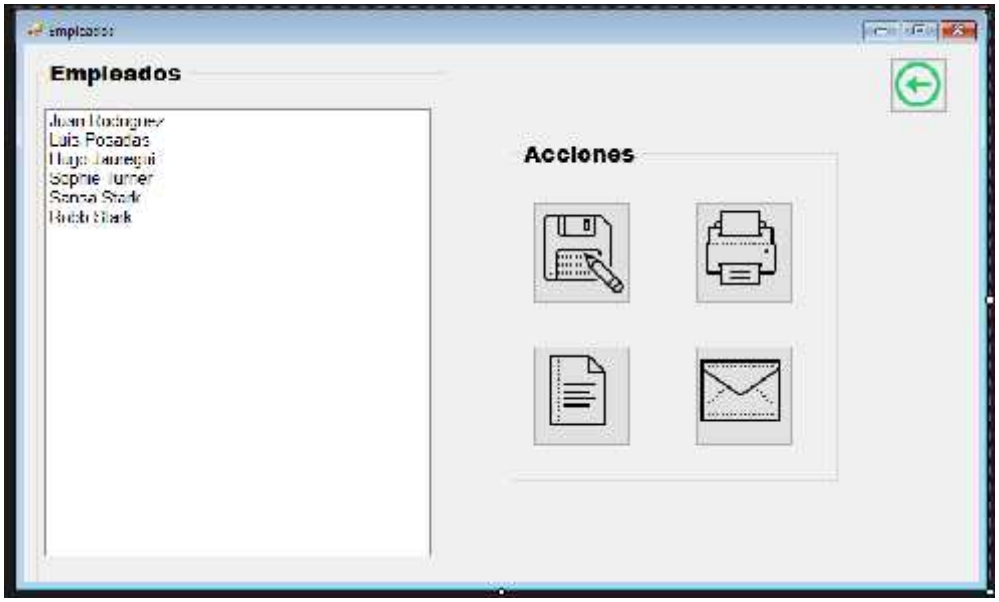
Un sistema de control está definido como un conjunto de componentes que pueden regular su propia conducta o la de otro sistema con el fin de lograr un funcionamiento predeterminado. Cabe resaltar que un sistema de control posee las características de: tener señal de corriente de entrada, Señal de corriente de salida, Variables manipuladas, variable controladora, conversiones, variaciones externas, fuentes de energía, retroalimentación.

Clasificación según su comportamiento:

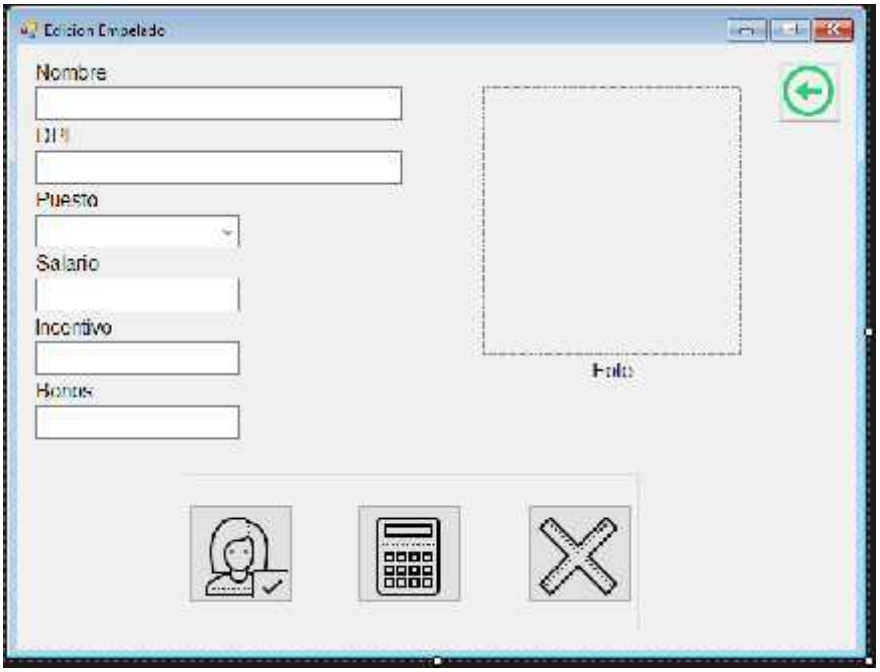
1. Sistema de Control de lazo abierto: Es aquel sistema que solo actúa el proceso sobre la señal de entrada y da como resultado una señal independiente. Estos sistemas se caracterizan por: Ser sencillos y de fácil conceptos, no tiene seguridad ante un fallo, la salida no se compara con la entrada, la precisión depende de la calibración.
2. Sistemas de control de lazo cerrado: Son los sistemas en los que la acción de control está en función de la señal de salida. Sus características son: Complejos pero amplios en parámetros, la salida se compara con la entrada, tienen la propiedad de retroalimentación, más estable al momento de errores.

INTERFAZ GRAFICA:

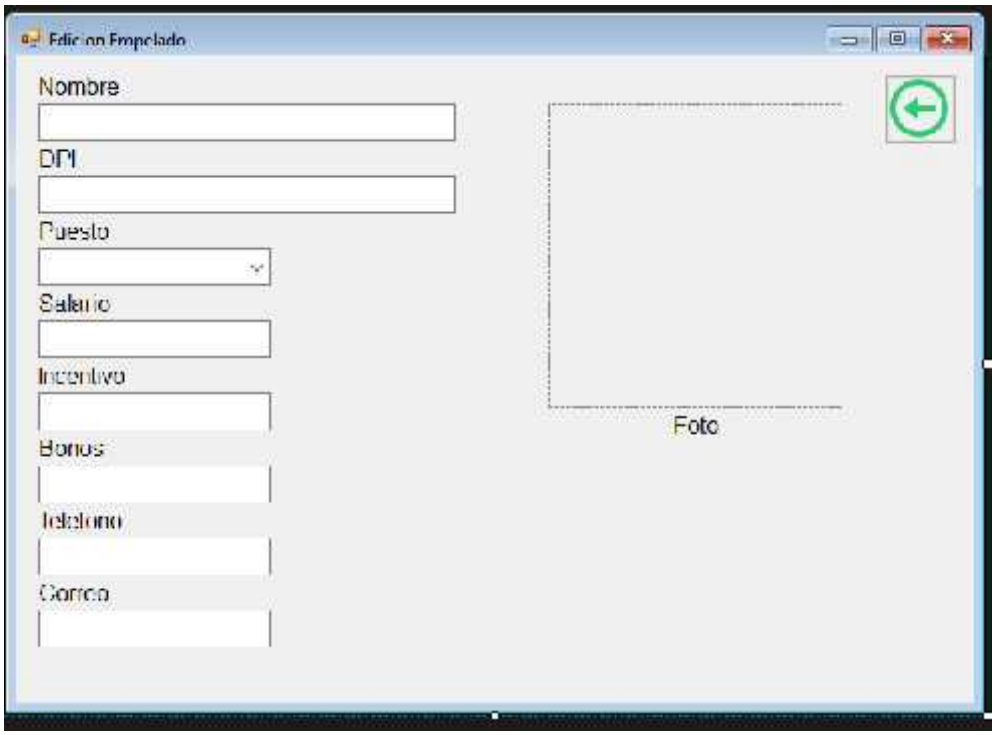
Ventana de inicio de sesión. Recordemos que cada usuario tiene su contraseña por lo tanto los administradores también tendrán un usuario y únicamente ellos podrán operar el sistema.



Ventana de sirve para el ingreso de empleados y nos permite realizar acciones como guardar los cambios en sueldos e incentivos, imprimir los datos, ver la ficha del empleado y contactar directamente con él también vemos el botón de regresar.



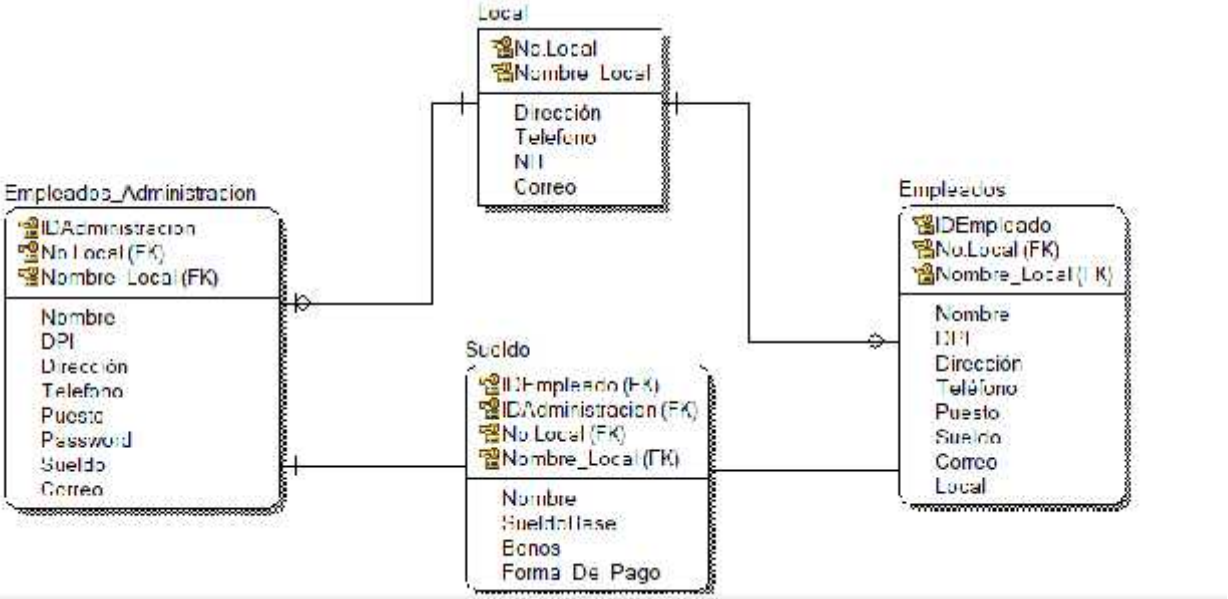
Tenemos la ventana de edición de empleado la cual nos permite editar datos como el puesto y salario del empleado, el botón de confirmación y el botón de cálculo de incentivo, además de un botón que nos permita cancelar los cambios realizados.



Y por último la ventana de ficha de usuario que nos muestra todos los datos del empleado, seleccionado.

HERRAMIENTA CASE:

En este fragmento veremos el desarrollo de nuestra herramienta case seleccionada Erwin la cual al realizar el diagrama de clases. Nos genera el script necesario para MySQL



Script:

```
CREATE TABLE Almacen
(
	Numero		char(18)  NOT NULL ,
	Dirección	char(18)  NULL ,
	Telefono	char(18)  NULL ,
	NIT		char(18)  NULL ,
	Correo		char(18)  NULL ,
	Nombre_Almacen	char(18)  NOT NULL
)
go


ALTER TABLE Almacen
	ADD CONSTRAINT XPKLocal PRIMARY KEY  CLUSTERED (Numero ASC,Nombre_Almacen ASC)
go


CREATE TABLE Empleados
(
	IDEmpleado	integer  NOT NULL ,
	Nombre		varchar(25)  NOT NULL ,
	DPI		varchar(20)  NOT NULL ,
	Dirección	varchar(20)  NULL ,
	Teléfono	varchar(9)  NOT NULL ,
	Puesto		varchar(20)  NOT NULL ,
	Sueldo		money  NOT NULL ,
	Correo		varchar(25)  NULL ,
	Numero		char(18)  NOT NULL ,
	Nombre_Almacen	char(18)  NOT NULL
)
go


ALTER TABLE Empleados
	ADD CONSTRAINT XPKEmpleados PRIMARY KEY CLUSTERED (IDEmpleado ASC,Numero
ASC,Nombre_Almacen ASC)
go


CREATE TABLE Empleados_Administracion
(
	IDAdministracion	integer  NOT NULL ,
	Nombre		varchar(22)  NOT NULL ,
	DPI		integer  NOT NULL ,
	Dirección	varchar(20)  NULL ,
	Telefono	varchar(9)  NOT NULL ,
	Puesto		varchar(20)  NOT NULL ,
	Password	varchar(12)  NOT NULL ,
	Sueldo		money  NOT NULL ,
	Correo		varchar(25)  NOT NULL ,
	Numero		char(18)  NOT NULL ,
	Nombre_Almacen	char(18)  NOT NULL
)
go


ALTER TABLE Empleados_Administracion
	ADD CONSTRAINT XPKEmpleados_Administracion PRIMARY KEY CLUSTERED (IDAdministracion
ASC,Numero ASC,Nombre_Almacen ASC)
go
```

```
CREATE TABLE Sueldo
(
        Nombre              varchar(25)  NOT NULL ,
        SueldoBase          money  NOT NULL ,
        Bonos               money  NOT NULL ,
        Forma_De_Pago       binary  NOT NULL ,
        IDEmpleado          integer  NOT NULL ,
        IDAdministracion    integer  NOT NULL ,
        Numero              char(18)  NOT NULL ,
        Nombre_Almacen      char(18)  NOT NULL
)
go


ALTER TABLE Sueldo
        ADD CONSTRAINT XPKSueldo PRIMARY KEY  CLUSTERED (IDEmpleado ASC,IDAdministracion ASC,Numero
ASC,Nombre_Almacen ASC)
go



ALTER TABLE Empleados
        ADD CONSTRAINT R_4 FOREIGN KEY (Numero,Nombre_Almacen) REFERENCES
Almacen(Numero,Nombre_Almacen)
                ON DELETE NO ACTION
                ON UPDATE NO ACTION
go



ALTER TABLE Empleados_Administracion
        ADD CONSTRAINT R_3 FOREIGN KEY (Numero,Nombre_Almacen) REFERENCES
Almacen(Numero,Nombre_Almacen)
                ON DELETE NO ACTION
                ON UPDATE NO ACTION
go



ALTER TABLE Sueldo
        ADD CONSTRAINT R_1 FOREIGN KEY (IDEmpleado,Numero,Nombre_Almacen) REFERENCES
Empleados(IDEmpleado,Numero,Nombre_Almacen)
                ON DELETE NO ACTION
                ON UPDATE NO ACTION
go



ALTER TABLE Sueldo
        ADD CONSTRAINT R_2 FOREIGN KEY (IDAdministracion,Numero,Nombre_Almacen) REFERENCES
Empleados_Administracion(IDAdministracion,Numero,Nombre_Almacen)
                ON DELETE NO ACTION
                ON UPDATE NO ACTION
go



CREATE TRIGGER tD_Almacen ON Almacen FOR DELETE AS
/* ERwin Builtin Trigger */
/* DELETE trigger on Almacen */
BEGIN
  DECLARE  @errno   int,
```

```
        @errmsg  varchar(255)
    /* ERwin Builtin Trigger */
    /* Almacen  Empleados_Administracion on parent delete no action */
    /* ERWIN_RELATION:CHECKSUM="000238eb", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados_Administracion"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_3", FK_COLUMNS="Numero""Nombre_Almacen" */
    IF EXISTS (
      SELECT * FROM deleted,Empleados_Administracion
      WHERE
        /*  %JoinFKPK(Empleados_Administracion,deleted," = "," AND") */
        Empleados_Administracion.Numero = deleted.Numero AND
        Empleados_Administracion.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30001,
            @errmsg = 'Cannot delete Almacen because Empleados_Administracion exists.'
      GOTO ERROR
    END

    /* ERwin Builtin Trigger */
    /* Almacen  Empleados on parent delete no action */
    /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_4", FK_COLUMNS="Numero""Nombre_Almacen" */
    IF EXISTS (
      SELECT * FROM deleted,Empleados
      WHERE
        /*  %JoinFKPK(Empleados,deleted," = "," AND") */
        Empleados.Numero = deleted.Numero AND
        Empleados.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30001,
            @errmsg = 'Cannot delete Almacen because Empleados exists.'
      GOTO ERROR
    END


    /* ERwin Builtin Trigger */
    RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go


CREATE TRIGGER tU_Almacen ON Almacen FOR UPDATE AS
/* ERwin Builtin Trigger */
/* UPDATE trigger on Almacen */
BEGIN
  DECLARE  @NUMROWS int,
           @nullcnt int,
           @validcnt int,
           @insNumero char(18),
           @insNombre_Almacen char(18),
           @errno   int,
           @errmsg  varchar(255)

  SELECT @NUMROWS = @@rowcount
  /* ERwin Builtin Trigger */
  /* Almacen  Empleados_Administracion on parent update no action */
  /* ERWIN_RELATION:CHECKSUM="00027d7e", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados_Administracion"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_3", FK_COLUMNS="Numero""Nombre_Almacen" */
  IF
```

```
    /* %ParentPK(" OR",UPDATE) */
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    IF EXISTS (
      SELECT * FROM deleted,Empleados_Administracion
      WHERE
        /*  %JoinFKPK(Empleados_Administracion,deleted," = "," AND") */
        Empleados_Administracion.Numero = deleted.Numero AND
        Empleados_Administracion.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30005,
            @errmsg = 'Cannot update Almacen because Empleados_Administracion exists.'
      GOTO ERROR
    END
  END

  /* ERwin Builtin Trigger */
  /* Almacen  Empleados on parent update no action */
  /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_4", FK_COLUMNS="Numero""Nombre_Almacen" */
  IF
    /* %ParentPK(" OR",UPDATE) */
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    IF EXISTS (
      SELECT * FROM deleted,Empleados
      WHERE
        /*  %JoinFKPK(Empleados,deleted," = "," AND") */
        Empleados.Numero = deleted.Numero AND
        Empleados.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30005,
            @errmsg = 'Cannot update Almacen because Empleados exists.'
      GOTO ERROR
    END
  END


  /* ERwin Builtin Trigger */
  RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go



CREATE TRIGGER tD_Empleados ON Empleados FOR DELETE AS
/* ERwin Builtin Trigger */
/* DELETE trigger on Empleados */
BEGIN
  DECLARE  @errno   int,
           @errmsg  varchar(255)
    /* ERwin Builtin Trigger */
    /* Empleados  Sueldo on parent delete no action */
    /* ERWIN_RELATION:CHECKSUM="000274c2", PARENT_OWNER="", PARENT_TABLE="Empleados"
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_1", FK_COLUMNS="IDEmpleado""Numero""Nombre_Almacen" */
    IF EXISTS (
      SELECT * FROM deleted,Sueldo
```

```
    WHERE
      /*  %JoinFKPK(Sueldo,deleted," = "," AND") */
      Sueldo.IDEmpleado = deleted.IDEmpleado AND
      Sueldo.Numero = deleted.Numero AND
      Sueldo.Nombre_Almacen = deleted.Nombre_Almacen
  )
  BEGIN
    SELECT @errno  = 30001,
          @errmsg = 'Cannot delete Empleados because Sueldo exists.'
    GOTO ERROR
  END

  /* ERwin Builtin Trigger */
  /* Almacen  Empleados on child delete no action */
  /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Almacen"
  CHILD_OWNER="", CHILD_TABLE="Empleados"
  P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
  FK_CONSTRAINT="R_4", FK_COLUMNS="Numero""Nombre_Almacen" */
  IF EXISTS (SELECT * FROM deleted,Almacen
    WHERE
      /* %JoinFKPK(deleted,Almacen," = "," AND") */
      deleted.Numero = Almacen.Numero AND
      deleted.Nombre_Almacen = Almacen.Nombre_Almacen AND
      NOT EXISTS (
        SELECT * FROM Empleados
        WHERE
          /* %JoinFKPK(Empleados,Almacen," = "," AND") */
          Empleados.Numero = Almacen.Numero AND
          Empleados.Nombre_Almacen = Almacen.Nombre_Almacen
      )
  )
  BEGIN
    SELECT @errno  = 30010,
          @errmsg = 'Cannot delete last Empleados because Almacen exists.'
    GOTO ERROR
  END


  /* ERwin Builtin Trigger */
  RETURN
ERROR:
  raiserror @errno @errmsg
  rollback transaction
END

go


CREATE TRIGGER tU_Empleados ON Empleados FOR UPDATE AS
/* ERwin Builtin Trigger */
/* UPDATE trigger on Empleados */
BEGIN
  DECLARE  @NUMROWS int,
          @nullcnt int,
          @validcnt int,
          @insIDEmpleado integer,
          @insNumero char(18),
          @insNombre_Almacen char(18),
          @errno   int,
          @errmsg  varchar(255)

  SELECT @NUMROWS = @@rowcount
  /* ERwin Builtin Trigger */
  /* Empleados  Sueldo on parent update no action */
  /* ERWIN_RELATION:CHECKSUM="0002aeaf", PARENT_OWNER="", PARENT_TABLE="Empleados"
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_1", FK_COLUMNS="IDEmpleado""Numero""Nombre_Almacen" */
  IF
    /* %ParentPK(" OR",UPDATE) */
```

```
    UPDATE(IDEmpleado) OR
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    IF EXISTS (
      SELECT * FROM deleted,Sueldo
      WHERE
        /*  %JoinFKPK(Sueldo,deleted," = "," AND") */
        Sueldo.IDEmpleado = deleted.IDEmpleado AND
        Sueldo.Numero = deleted.Numero AND
        Sueldo.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30005,
            @errmsg = 'Cannot update Empleados because Sueldo exists.'
      GOTO ERROR
    END
  END

  /* ERwin Builtin Trigger */
  /* Almacen  Empleados on child update no action */
  /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_4", FK_COLUMNS="Numero""Nombre_Almacen" */
  IF
    /* %ChildFK(" OR",UPDATE) */
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    SELECT @nullcnt = 0
    SELECT @validcnt = count(*)
      FROM inserted,Almacen
        WHERE
          /* %JoinFKPK(inserted,Almacen) */
          inserted.Numero = Almacen.Numero and
          inserted.Nombre_Almacen = Almacen.Nombre_Almacen
    /* %NotnullFK(inserted," IS NULL","select @nullcnt = count(*) from inserted where"," AND") */

    IF @validcnt + @nullcnt != @NUMROWS
    BEGIN
      SELECT @errno  = 30007,
            @errmsg = 'Cannot update Empleados because Almacen does not exist.'
      GOTO ERROR
    END
  END


  /* ERwin Builtin Trigger */
  RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go



CREATE TRIGGER tD_Empleados_Administracion ON Empleados_Administracion FOR DELETE AS
/* ERwin Builtin Trigger */
/* DELETE trigger on Empleados_Administracion */
BEGIN
  DECLARE  @errno   int,
           @errmsg  varchar(255)
    /* ERwin Builtin Trigger */
    /* Empleados_Administracion  Sueldo on parent delete no action */
    /*                              ERWIN_RELATION:CHECKSUM="0002ad05",                              PARENT_OWNER="",
PARENT_TABLE="Empleados_Administracion"
```

```
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_2", FK_COLUMNS="IDAdministracion""Numero""Nombre_Almacen" */
    IF EXISTS (
      SELECT * FROM deleted,Sueldo
      WHERE
        /*  %JoinFKPK(Sueldo,deleted," = "," AND") */
        Sueldo.IDAdministracion = deleted.IDAdministracion AND
        Sueldo.Numero = deleted.Numero AND
        Sueldo.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30001,
            @errmsg = 'Cannot delete Empleados_Administracion because Sueldo exists.'
      GOTO ERROR
    END

    /* ERwin Builtin Trigger */
    /* Almacen  Empleados_Administracion on child delete no action */
    /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados_Administracion"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_3", FK_COLUMNS="Numero""Nombre_Almacen" */
    IF EXISTS (SELECT * FROM deleted,Almacen
      WHERE
        /* %JoinFKPK(deleted,Almacen," = "," AND") */
        deleted.Numero = Almacen.Numero AND
        deleted.Nombre_Almacen = Almacen.Nombre_Almacen AND
        NOT EXISTS (
          SELECT * FROM Empleados_Administracion
          WHERE
            /* %JoinFKPK(Empleados_Administracion,Almacen," = "," AND") */
            Empleados_Administracion.Numero = Almacen.Numero AND
            Empleados_Administracion.Nombre_Almacen = Almacen.Nombre_Almacen
        )
    )
    BEGIN
      SELECT @errno  = 30010,
            @errmsg = 'Cannot delete last Empleados_Administracion because Almacen exists.'
      GOTO ERROR
    END


    /* ERwin Builtin Trigger */
    RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go


CREATE TRIGGER tU_Empleados_Administracion ON Empleados_Administracion FOR UPDATE AS
/* ERwin Builtin Trigger */
/* UPDATE trigger on Empleados_Administracion */
BEGIN
  DECLARE  @NUMROWS int,
           @nullcnt int,
           @validcnt int,
           @insIDAdministracion integer,
           @insNumero char(18),
           @insNombre_Almacen char(18),
           @errno   int,
           @errmsg  varchar(255)

  SELECT @NUMROWS = @@rowcount
  /* ERwin Builtin Trigger */
  /* Empleados_Administracion  Sueldo on parent update no action */
  /* ERWIN_RELATION:CHECKSUM="0002f69e", PARENT_OWNER="", PARENT_TABLE="Empleados_Administracion"
```

```
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_2", FK_COLUMNS="IDAdministracion""Numero""Nombre_Almacen" */
  IF
    /* %ParentPK(" OR",UPDATE) */
    UPDATE(IDAdministracion) OR
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    IF EXISTS (
      SELECT * FROM deleted,Sueldo
      WHERE
        /*  %JoinFKPK(Sueldo,deleted," = "," AND") */
        Sueldo.IDAdministracion = deleted.IDAdministracion AND
        Sueldo.Numero = deleted.Numero AND
        Sueldo.Nombre_Almacen = deleted.Nombre_Almacen
    )
    BEGIN
      SELECT @errno  = 30005,
            @errmsg = 'Cannot update Empleados_Administracion because Sueldo exists.'
      GOTO ERROR
    END
  END

  /* ERwin Builtin Trigger */
  /* Almacen  Empleados_Administracion on child update no action */
  /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Almacen"
    CHILD_OWNER="", CHILD_TABLE="Empleados_Administracion"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_3", FK_COLUMNS="Numero""Nombre_Almacen" */
  IF
    /* %ChildFK(" OR",UPDATE) */
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    SELECT @nullcnt = 0
    SELECT @validcnt = count(*)
      FROM inserted,Almacen
        WHERE
          /* %JoinFKPK(inserted,Almacen) */
          inserted.Numero = Almacen.Numero and
          inserted.Nombre_Almacen = Almacen.Nombre_Almacen
    /* %NotnullFK(inserted," IS NULL","select @nullcnt = count(*) from inserted where"," AND") */

    IF @validcnt + @nullcnt != @NUMROWS
    BEGIN
      SELECT @errno  = 30007,
            @errmsg = 'Cannot update Empleados_Administracion because Almacen does not exist.'
      GOTO ERROR
    END
  END


  /* ERwin Builtin Trigger */
  RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go



CREATE TRIGGER tD_Sueldo ON Sueldo FOR DELETE AS
/* ERwin Builtin Trigger */
/* DELETE trigger on Sueldo */
BEGIN
  DECLARE  @errno   int,
```

```
        @errmsg  varchar(255)
    /* ERwin Builtin Trigger */
    /* Empleados  Sueldo on child delete no action */
    /* ERWIN_RELATION:CHECKSUM="00038de3", PARENT_OWNER="", PARENT_TABLE="Empleados"
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_1", FK_COLUMNS="IDEmpleado""Numero""Nombre_Almacen" */
    IF EXISTS (SELECT * FROM deleted,Empleados
      WHERE
        /* %JoinFKPK(deleted,Empleados," = "," AND") */
        deleted.IDEmpleado = Empleados.IDEmpleado AND
        deleted.Numero = Empleados.Numero AND
        deleted.Nombre_Almacen = Empleados.Nombre_Almacen AND
        NOT EXISTS (
          SELECT * FROM Sueldo
          WHERE
            /* %JoinFKPK(Sueldo,Empleados," = "," AND") */
            Sueldo.IDEmpleado = Empleados.IDEmpleado AND
            Sueldo.Numero = Empleados.Numero AND
            Sueldo.Nombre_Almacen = Empleados.Nombre_Almacen
        )
    )
    BEGIN
      SELECT @errno  = 30010,
          @errmsg = 'Cannot delete last Sueldo because Empleados exists.'
      GOTO ERROR
    END

    /* ERwin Builtin Trigger */
    /* Empleados_Administracion  Sueldo on child delete no action */
    /*                              ERWIN_RELATION:CHECKSUM="00000000",                              PARENT_OWNER="",
PARENT_TABLE="Empleados_Administracion"
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_2", FK_COLUMNS="IDAdministracion""Numero""Nombre_Almacen" */
    IF EXISTS (SELECT * FROM deleted,Empleados_Administracion
      WHERE
        /* %JoinFKPK(deleted,Empleados_Administracion," = "," AND") */
        deleted.IDAdministracion = Empleados_Administracion.IDAdministracion AND
        deleted.Numero = Empleados_Administracion.Numero AND
        deleted.Nombre_Almacen = Empleados_Administracion.Nombre_Almacen AND
        NOT EXISTS (
          SELECT * FROM Sueldo
          WHERE
            /* %JoinFKPK(Sueldo,Empleados_Administracion," = "," AND") */
            Sueldo.IDAdministracion = Empleados_Administracion.IDAdministracion AND
            Sueldo.Numero = Empleados_Administracion.Numero AND
            Sueldo.Nombre_Almacen = Empleados_Administracion.Nombre_Almacen
        )
    )
    BEGIN
      SELECT @errno  = 30010,
          @errmsg = 'Cannot delete last Sueldo because Empleados_Administracion exists.'
      GOTO ERROR
    END


    /* ERwin Builtin Trigger */
    RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go


CREATE TRIGGER tU_Sueldo ON Sueldo FOR UPDATE AS
/* ERwin Builtin Trigger */
/* UPDATE trigger on Sueldo */
```

```
BEGIN
  DECLARE  @NUMROWS int,
           @nullcnt int,
           @validcnt int,
           @insIDEmpleado integer,
           @insIDAdministracion integer,
           @insNumero char(18),
           @insNombre_Almacen char(18),
           @errno   int,
           @errmsg  varchar(255)

  SELECT @NUMROWS = @@rowcount
  /* ERwin Builtin Trigger */
  /* Empleados  Sueldo on child update no action */
  /* ERWIN_RELATION:CHECKSUM="00038090", PARENT_OWNER="", PARENT_TABLE="Empleados"
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_1", FK_COLUMNS="IDEmpleado""Numero""Nombre_Almacen" */
  IF
    /* %ChildFK(" OR",UPDATE) */
    UPDATE(IDEmpleado) OR
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    SELECT @nullcnt = 0
    SELECT @validcnt = count(*)
      FROM inserted,Empleados
        WHERE
          /* %JoinFKPK(inserted,Empleados) */
          inserted.IDEmpleado = Empleados.IDEmpleado and
          inserted.Numero = Empleados.Numero and
          inserted.Nombre_Almacen = Empleados.Nombre_Almacen
    /* %NotnullFK(inserted," IS NULL","select @nullcnt = count(*) from inserted where"," AND") */

    IF @validcnt + @nullcnt != @NUMROWS
    BEGIN
      SELECT @errno  = 30007,
             @errmsg = 'Cannot update Sueldo because Empleados does not exist.'
      GOTO ERROR
    END
  END

  /* ERwin Builtin Trigger */
  /* Empleados_Administracion  Sueldo on child update no action */
  /* ERWIN_RELATION:CHECKSUM="00000000", PARENT_OWNER="", PARENT_TABLE="Empleados_Administracion"
    CHILD_OWNER="", CHILD_TABLE="Sueldo"
    P2C_VERB_PHRASE="", C2P_VERB_PHRASE="",
    FK_CONSTRAINT="R_2", FK_COLUMNS="IDAdministracion""Numero""Nombre_Almacen" */
  IF
    /* %ChildFK(" OR",UPDATE) */
    UPDATE(IDAdministracion) OR
    UPDATE(Numero) OR
    UPDATE(Nombre_Almacen)
  BEGIN
    SELECT @nullcnt = 0
    SELECT @validcnt = count(*)
      FROM inserted,Empleados_Administracion
        WHERE
          /* %JoinFKPK(inserted,Empleados_Administracion) */
          inserted.IDAdministracion = Empleados_Administracion.IDAdministracion and
          inserted.Numero = Empleados_Administracion.Numero and
          inserted.Nombre_Almacen = Empleados_Administracion.Nombre_Almacen
    /* %NotnullFK(inserted," IS NULL","select @nullcnt = count(*) from inserted where"," AND") */

    IF @validcnt + @nullcnt != @NUMROWS
    BEGIN
      SELECT @errno  = 30007,
             @errmsg = 'Cannot update Sueldo because Empleados_Administracion does not exist.'
      GOTO ERROR
    END
```

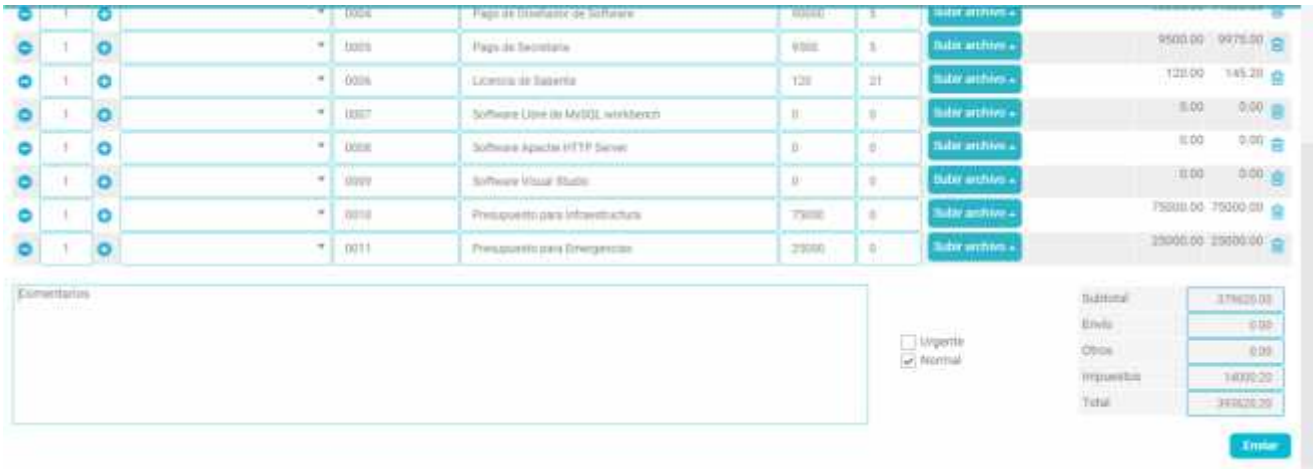
```
  END


  /* ERwin Builtin Trigger */
  RETURN
ERROR:
    raiserror @errno @errmsg
    rollback transaction
END

go
```

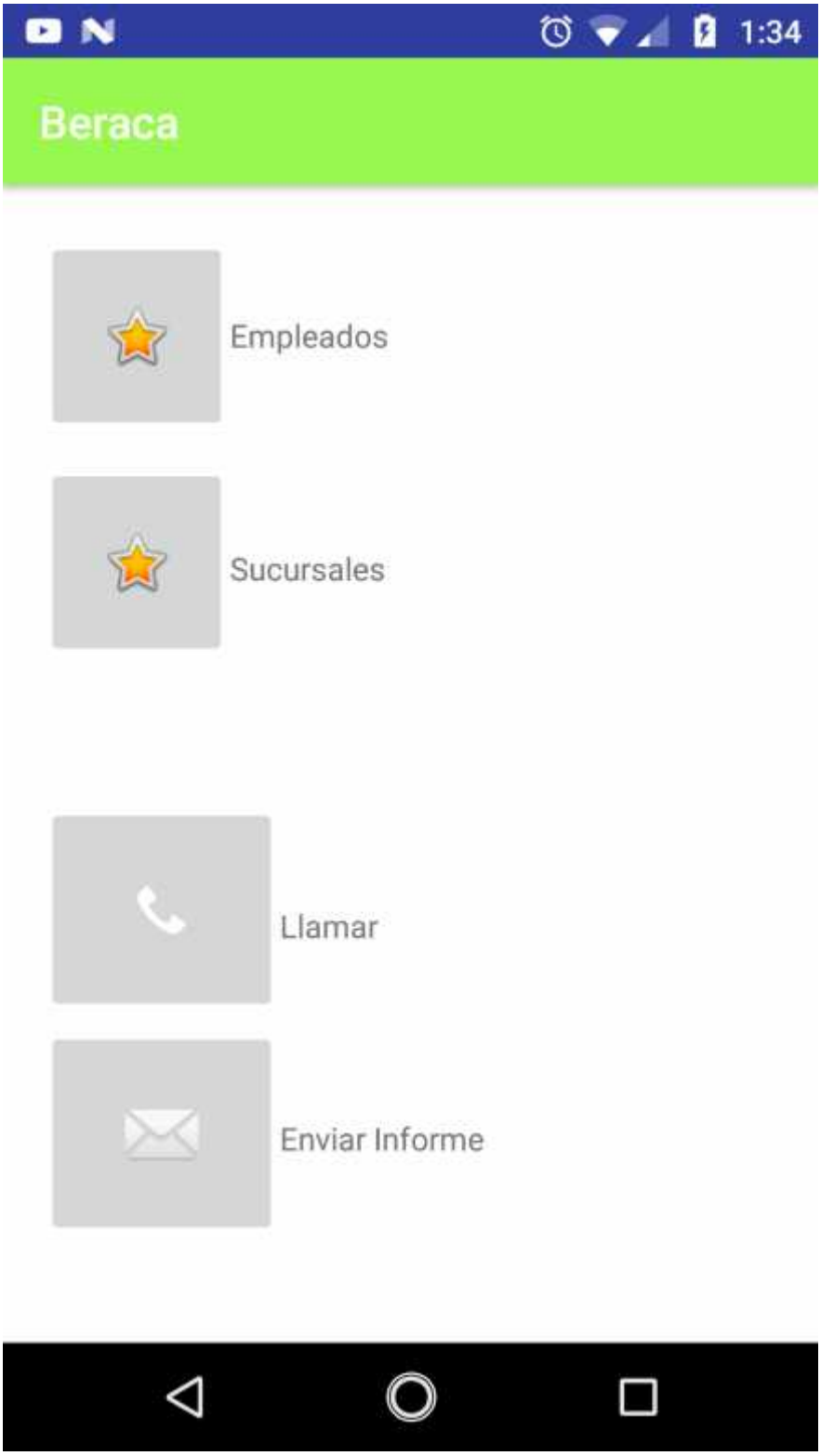
PRESUPUESTO:

Actualmente el proyecto está determinado en un presupuesto de Q.393, 620.20 aprovechado gran cantidad de softwares libres como apache y MySQL workbench para ahorrar todo lo posible en licencias. El mayor problema de fue un cálculo aproximado del sueldo por mes de un desarrollador, que tras varias consultas se determinó que una cantidad aceptable era Q20, 000 al mes. Gracias a Sapenta se logró un desglose claro.



Por parte de la empresa seria dada una capacitación a los empleados para el manejo de software y un mantenimiento trimestral de Q5,000 para mantener un óptimo rendimiento. Por al menos dos años.

Interfaz de Celular:




8:00
Nombre
Name
Puesto
Puesto
salario
Salario
Bono
Bono
GUARDAR
BUSCAR
BORRAR