UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN, CONTROL Y EVALUACIÓN DE RECURSOS INFORMÁTICOS

INGENIERÍA DE SISTEMAS DE INFORMACIÓN

"IMPLEMENTACIÓN DE UNA BASE DE DATOS PARA EL PROCESO DE DISTRIBUCIÓN DE DONACIONES A HOSPITALES"

ASIGNATURA:

SISTEMAS DE BASE DE DATOS II

ELABORADO POR: CAMPOS, MILAGROS 8-948-227

CASTILLO GUADALUPE 8-929-2252

ROSAS, ELIONAYS 9-756-2182

URRIOLA, ALEJANDRO 9-755-1141

FACILITADOR: ING. HENRY LEZCANO

II SEMESTRE, 2020

Contenido

Introducción 3
Misión de la Base de Datos 4
Objetivo de la misión de la Base de Datos 4
Definición del proyecto 5
Descripción de los datos que han sido identificados y utilización 6
Modelado de la Base de Datos 7
MODELO LOGICO NORMALIZADO 8
MODELO FÍSICO EN ORACLE 10
CREACION DE TABLAS 10
PROCEDIMIENTOS UTILIZADOS 22
IMPLEMENTACION DE TRIGGERS 44
CONCLUSIÓN 51

Introducción

En 2020, Panamá fue afectada terriblemente por el brote de covid-2019 este ha hecho que la organización y sistema de salud en general colapse, generando un desabastecimiento de los insumos médicos en los diferentes centros de salud y hospitales, por ello surge la necesidad de buscar un mecanismo mediante el cual optimizar los procesos de distribución de los insumos médicos, y así evitar el desabastecimiento de las instalaciones médicas. Para lograr esto hemos planteado la creación de una base de datos, de la cual mas adelante iremos viendo cada uno de sus aspectos mas importantes, desde su modelo lógico, modelo relacional y la implementación de algunos datos de prueba.

Nombre del proyecto de Base de Datos

IMPLEMENTACIÓN DE UNA BASE DE DATOS PARA EL PROCESO DE DISTRIBUCIÓN DE DONACIONES A HOSPITALES

Misión de la Base de Datos

Lograr que las donaciones que realiza el pueblo panameño mediante el Plan Panamá solidario se distribuyan equitativamente a todos los centros médicos del país que se encuentran atendiendo a pacientes positivos de covid-19.

Objetivo de la misión de la Base de Datos

- ✓ Mantener el control de las donaciones de insumos médicos llevadas a los distintos centros de distribución para evitar la mala disposición de estas.
- ✓ Proveer información de los diversos tipos de transporte utilizados para gestión de la distribución de los insumos médicos a las distintas instalaciones médicas.
- ✓ Sostener un adecuado manejo del personal que trabaja en los distintos centros de distribución.

Definición del proyecto

- Ámbito

  Actualmente el plan Panamá solidario en la sección de donaciones de salud se maneja con una base de datos que recopila información del donante y de la donación, lo que incluye todos los datos que describen el insumo, que llegan a los centros de distribución que se encuentran ubicados en distintas regiones del país. Destacamos que el sistema actual permite saber las necesidades urgentes que tiene el sector salud en cuanto a los insumos médicos, sin embargo, presenta puntos de mejora en el control de las donaciones, por lo que esta base de datos pretende solucionar esta problemática de manera efectiva proporcionando así agilidad en todos los procesos que sean necesarios para el desenvolvimiento óptimo de la distribución de insumos.

- Alcance o limites

  - ✓ Nuestra base de datos debe ser capaz de almacenar la información del donante, insumos médicos (medicinas y equipo de protección), los distintos centros de distribución que se encuentran en diversas regiones del país, el personal que este posee (empacador y repartidor) y el transporte utilizado para la distribución.

  - ✓ Debe permitir ingresar, eliminar y actualizar la información de todos los insumos médicos recibidos en los diversos centros de distribución.

- Análisis de requerimientos

Para la recopilación de datos de este proyecto decidimos utilizar la técnica de Revisión de documentos obtenidos del sitio web oficial del programa Panamá Solidario, de esta manera aseguramos la comprensión del surgimiento de la necesidad del nuevo sistema haciendo uso minucioso de cada dato almacenado que guardaba relación con el problema que se debía resolver adicional a eso realizamos algunas consultas a manera de nutrir al equipo de información importante sobre la situación actual en Panamá y el mundo por lo promovemos la obtención de información desde una vista mucho más amplia que la que se tenía actualmente asegurando la implementación de los mejores métodos o métodos más funcionales en nuestro sistema de base de datos

Descripción de los datos que han sido identificados y utilización

Al tratarse de un proyecto basado un poco en el tema logístico los datos tienden a ser genéricos, pero a más detalle se puede tener una mejor toma de decisiones por lo que cada tabla cuenta con datos indispensables.

En la tabla Donador, manejas los datos personales de cada persona o empresa que hace una donación, se hace indispensable su dirección para de esta manera conocer los tiempos de llegada de las donaciones o de donde provienen.

En la tabla Centro de Distribución, manejamos todos los datos de ubicación entre ellos la región en que está ubicada de esta manera los usuarios de la base de datos pueden saber todas las áreas que pueden ser atendidas por este de igual manera para la tabla instalaciones médicas donde se detalla la región donde está ubicada.

Sabemos también del personal que labora en los centros de distribución su fecha de nacimiento y su dirección, para asegurar que no se encuentre en un rango de edad en alto riesgo y donde puede ser enviado como colaborador respectivamente relacionado al riesgo, pero en otro contexto tenemos también la tabla transporte donde se maneja la capacidad de carga de cada vehículo asegurando el uso óptimo de cada uno.

Modelado de la Base de Datos

- Modelo conceptual E/R

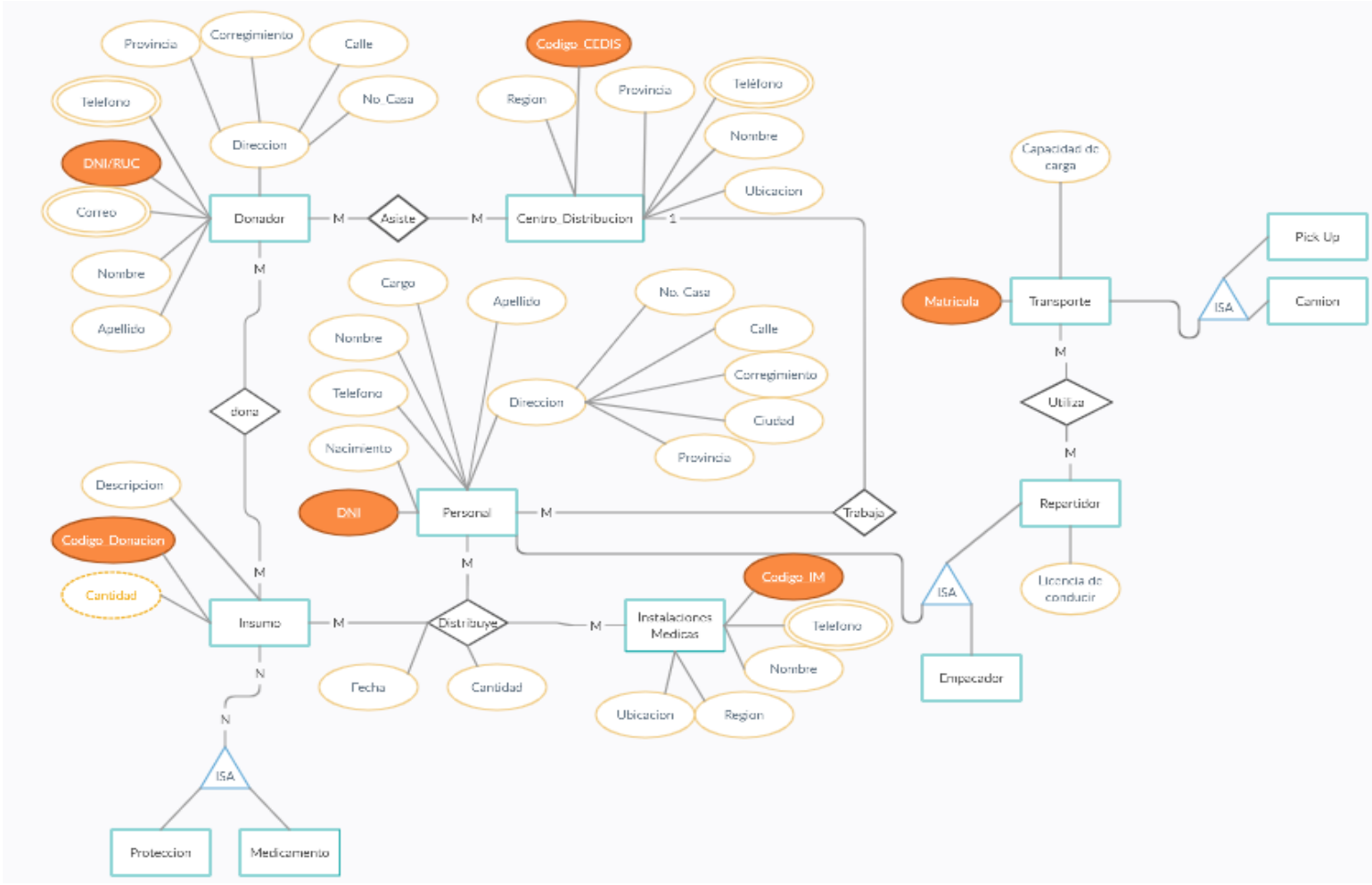


Ilustración 1 Modelo conceptual E/R

MODELO LOGICO NORMALIZADO

Centro_distribucion		
PK	codigo_CEDIS	not null
	nombre	not null
	region	not null
	provincia	not null
	ubicación	not null

tipo_telefono		
PK	cod_telf	not null
	desc_telf	

tipo_email		
PK	cod_email	not null
	desc_email	

telefono_donador		
PK/FK	DNI_RUC	not null
PK/FK	cod_telf	not null
	telefono	

Personal		
PK	DNI	not null
	nombre	not null
	apellido	not null
	fecha_nac	not null
	ciudad	not null
	corregimiento	not null
	calle	
	num_casa	not null
FK	cod_cargo	not null

telefono_personal		
PK/FK	DNI	not null
PK/FK	cod_telf	not null
	telefono	

email_donador		
PK/FK	DNI_RUC	not null
PK/FK	cod_email	not null
	email	

Donador		
PK	DNI_RUC	not null
	nombre	not null
	apellido	not null
	provincia	not null
	distrito	
	corregimiento	
	calle	
	num_casa	

tipo_insumo		
PK	cod_insumo	not null
	desc_insumo	

Insumo		
PK	cod_donacion	not null
FK	cod_insumo	not null

Instalaciones_medicas		
PK	codigo_IM	not null
	nombre	not null
	region	not null
	provincia	not null
	distrito	not null
	corregimiento	not null
	ubicación	not null

tipo_transporte		
PK	cod_trans	not null
	desc_trans	

telefono_IM		
PK/FK	codigo_IM	not null
PK/FK	cod_telf	not null
	telefono	

Transporte		
PK	matricula	not null
	cap_carga	not null
FK	cod_trans	not null

Repartidor		
PK	licencia	not null
FK	DNI	not null
FK	matricula	not null

Insumo_personal_IM		
PK/FK	cod_insumo	not null
PK/FK	DNI	not null
PK/FK	codigo_IM	not null
	fecha_dist	
	cantidad_llevada	

Donador_insumo		
PK/FK	DNI_RUC	not null
PK/FK	cod_donacion	not null
	Cantidad	Not null

Personal_CEDIS		
PK/FK	DNI	not null
PK/FK	codigo_CEDIS	not null

tipo_personal		
PK	cod_cargo	not null
	descripcion	

Donador_CEDIS		
PK/FK	DNI_RUC	not null
PK/FK	codigo_CEDIS	not null

MODELO FÍSICO EN ORACLE

CREACION DE TABLAS

--Creacion de la tabla TIPO_TELEFONO

```
Create table Tipo_Telefono (
        Cod_Telf varchar2(5) not null primary key,
        Desc_Telf varchar2(15) not null
 );
```

```
SQL> Create table Tipo_Telefono (
  2  Cod_Telf varchar2(5) not null primary key,
  3  Desc_Telf varchar2(15) not null
  4   );

Table created.
```

-- Creación de la tabla llamada DONADOR

```
Create table DONADOR (
        DNI_RUC  varchar(20) not null primary key,
        Nombre  varchar (10) not null,
        Apellido varchar (10) not null,
        provincia varchar(20) not null,
         Distrito varchar (30),
         Corregimiento varchar (30),
         calle varchar (20),
          no_casa varchar (5)
   );
```

```
SQL> Create table DONADOR (
  2    DNI_RUC  varchar(20) not null primary key,
  3             Nombre  varchar (10) not null,
  4               Apellido varchar (10) not null,
  5              provincia varchar(20) not null,
  6              Distrito varchar (30),
  7              Corregimiento varchar (30),
  8              calle varchar (20),
  9              no_casa varchar (5)
 10               );

Table created.
```

--Creacion de la tabla TELEFONO_DONADOR

```
Create table Telefono_donador (
        DNI_RUC varchar2(20) not null,
        Constraint dni_fk_1 foreign key (DNI_RUC) references Donador (DNI_RUC),
        Cod_Telf varchar2(5) not null,
        Constraint cod_Telef_fk foreign key (Cod_Telf) references Tipo_Telefono,
        Constraint Telefonos_donador primary key (DNI_RUC, Cod_telf)
);
```

```
SQL> Create table Telefono_donador (
  2  DNI_RUC varchar2(20) not null,
  3  Constraint dni_fk_1 foreign key (DNI_RUC) references Donador
(DNI_RUC),
  4  Cod_Telf varchar2(5) not null,
  5  Constraint cod_Telef_fk foreign key (Cod_Telf) references Tip
o_Telefono,
  6  Constraint Telefonos_donador primary key (DNI_RUC, Cod_telf)
  7  );

Table created.
```

ALTER TABLE TELEFONO_DONADOR ADD TELEFONO NUMBER;

```
SQL> ALTER TABLE TELEFONO_DONADOR ADD TELEFONO NUMBER;

Table altered.
```

--Creacion de la tabla TIPO_EMAIL

```
Create table Tipo_Email (
        Cod_Email varchar2(5) not null primary key,
        Desc_Email varchar2(25) not null
 );
```

```
SQL> --Creacion de la tabla TIPO_EMAIL
SQL> Create table Tipo_Email (
  2  Cod_Email varchar2(5) not null primary key,
  3  Desc_Email varchar2(25) not null
  4   );

Table created.
```

--Creacion de la tabla EMAIL_donador

```
Create table Email_donador (
DNI_RUC varchar2(20),
Constraint dni_ruc_fk_2 foreign key (DNI_RUC) references Donador(DNI_RUC),
Cod_Email varchar2(5),
Constraint cod_email_fk foreign key (Cod_Email) references Tipo_Email (Cod_Email),
Constraint Emails_donador primary key (DNI_RUC, Cod_Email)
);
```

```
SQL> Create table Email_donador (
  2  DNI_RUC varchar2(20),
  3  Constraint dni_ruc_fk_2 foreign key (DNI_RUC) references Dona
dor(DNI_RUC),
  4  Cod_Email varchar2(5),
  5  Constraint cod_email_fk foreign key (Cod_Email) references Ti
po_Email (Cod_Email),
  6  Constraint Emails_donador primary key (DNI_RUC, Cod_Email)
  7  );

Table created.
```

ALTER TABLE EMAIL_DONADOR ADD EMAIL VARCHAR2(30);

```
SQL> ALTER TABLE EMAIL_DONADOR ADD EMAIL VARCHAR2(30);

Table altered.
```

--Creacion de la tabla Centro distribución

```
Create table Centro_distribucion(
Codigo_cedis varchar2(5) not null primary key,
Nombre varchar2(20) not null,
Region varchar2(15) not null,
Provincia varchar2(15) not null,
Ubicacion varchar2(20) not null
);
```

```
SQL> Create table Centro_distribucion(
  2  Codigo_cedis varchar2(5) not null primary key,
  3  Nombre varchar2(20) not null,
  4  Region varchar2(15) not null,
  5  Provincia varchar2(15) not null,
  6  Ubivcacion varchar2(20) not null
  7  );

Table created.
```

--Creacion de la tabla Tipo insumo

```
Create table Tipo_Insumo(
Cod_insumo varchar2(5) not null primary key,
Desc_insumo varchar2(20)
);
```

```
SQL> Create table Tipo_Insumo(
  2  Cod_insumo varchar2(5) not null primary key,
  3  Desc_insumo varchar2(20)
  4  );

Table created.
```

--Creacion de la tabla Insumo

```
Create table Insumo(
Cod_donacion number not null primary key,
Cantidad number not null,
Cod_insumo varchar2(5),
Constraint FK_cod_insumo foreign key(cod_insumo) references Tipo_insumo(cod_insumo
)
);
```

```
SQL> Create table Insumo(
  2  Cod_donacion number not null primary key,
  3  Cantidad number not null,
  4  Cod_insumo varchar2(5),
  5  Constraint FK_cod_insumo foreign key(cod_insumo) references T
ipo_insumo(cod_insumo)
  6  );

Table created.
```

ALTER TABLE INSUMO DROP COLUMN CANTIDAD;

```
SQL> ALTER TABLE INSUMO DROP COLUMN CANTIDAD;

Table altered.
```

ALTER TABLE INSUMO ADD Cantidad_total number;

```
SQL> ALTER TABLE INSUMO ADD Cantidad_total number;

Table altered.
```

ALTER TABLE INSUMO ADD DESCRIPCION VARCHAR2(20);

```
SQL> ALTER TABLE INSUMO ADD DESCRIPCION VARCHA
```

--Creacion de la tabla Instalaciones Medicas

```
Create table Instalaciones_medicas(
Codigo_IM varchar2(5) not null primary key,
Nombre varchar2(20) not null,
Region varchar2(15) not null,
Provincia varchar2(15) not null,
Distrito varchar2(15) not null,
Corregimiento varchar2(15) not null,
Ubicacion varchar2(20 ) not null
);
```

```
SQL> Create table Instalaciones_medicas(
  2  Codigo_IM varchar2(5) not null primary key,
  3  Nombre varchar2(20) not null,
  4  Region varchar2(15) not null,
  5  Provincia varchar2(15) not null,
  6  Distrito varchar2(15) not null,
  7  Corregimiento varchar2(15) not null,
  8  Ubicacion varchar2(20 ) not null
  9  );

Table created.
```

ALTER TABLE Instalaciones_medicas DROP COLUMN provincia;

ALTER TABLE Instalaciones_medicas DROP COLUMN corregimiento;

ALTER TABLE Instalaciones_medicas DROP COLUMN distrito;

```
SQL> ALTER TABLE Instalaciones_medicas DROP COLUMN provincia;

Table altered.

SQL> ALTER TABLE Instalaciones_medicas DROP COLUMN corregimiento;

Table altered.

SQL> ALTER TABLE Instalaciones_medicas DROP COLUMN distrito;

Table altered.
```

--Creacion de la tabla teléfono de Instalación medica

```
Create table telefono_im (
Codigo_IM varchar2(5) not null,
Constraint fk_codigo_im foreign key (codigo_IM) references Instalaciones_medicas(Codig
o_IM),
Cod_telf varchar2(5) not null,
Constraint fk_cod_telf foreign key (Cod_telf) references tipo_telefono(Cod_telf),
Constraint telefono_im primary key (codigo_IM,Cod_telf)
);
```

```
SQL> Create table telefono_im (
  2  Codigo_IM varchar2(5) not null,
  3  Constraint fk_codigo_im foreign key (codigo_IM) references In
stalaciones_medicas(Codigo_IM),
  4  Cod_telf varchar2(5) not null,
  5  Constraint fk_cod_telf foreign key (Cod_telf) references tipo
_telefono(Cod_telf),
  6  Constraint telefono_im primary key (codigo_IM,Cod_telf)
  7  );

Table created.
```

ALTER TABLE TELEFONO_IM ADD TELEFONO NUMBER;

```
SQL> ALTER TABLE TELEFONO_IM ADD TELEFONO NUMBER;

Table altered.
```

--Creacion de la tabla Tipo transporte

Create table tipo_transporte(
Cod_trans varchar2(5) not null primary key,
Desc_trans varchar(15)
);

```
SQL> Create table tipo_transporte(
  2  Cod_trans varchar2(5) not null primary key,
  3  Desc_trans varchar(15)
  4  );

Table created.
```

--Creacion de la tabla transporte

Create table transporte(
Matricula varchar2(10) not null primary key,
Cap_carga number not null,
Cod_trans varchar2(5),
Constraint fk_cos_trans foreign key (Cod_trans) references tipo_transporte (Cod_trans)
);

```
SQL> Create table transporte(
  2  Matricula varchar2(10) not null primary key,
  3  Cap_carga number not null,
  4  Cod_trans varchar2(5),
  5  Constraint fk_cos_trans foreign key (Cod_trans) references ti
po_transporte (Cod_trans)
  6  );

Table created.
```

--Creación de tabla Tipo personal

Create table tipo_personal(
Cod_cargo varchar2(5) not null primary key,
Descripcion varchar2(20)
);

```
SQL> Create table tipo_personal(
  2  Cod_cargo varchar2(5) not null primary key,
  3  Descripcion varchar2(20)
  4  );

Table created.
```

-- **Creación de tabla personal**

Create table Personal(
DNI varchar2(15) not null primary key,
Nombre varchar2(15) not null,
Apellido varchar2(15) not null,
Fecha_nac date not null,
Ciudad varchar2(15) not null,
Corregimiento varchar2(15) not null,
Calle varchar2(10),
Num_casa varchar2(5) not null,
Cod_cargo varchar2(5),
Constraint fk_cod_cargo foreign key (Cod_cargo) references tipo_personal(Cod_cargo)
);

```
SQL> Create table Personal(
  2  DNI varchar2(15) not null primary key,
  3  Nombre varchar2(15) not null,
  4  Apellido varchar2(15) not null,
  5  Fecha_nac date not null,
  6  Ciudad varchar2(15 ) not null,
  7  Corregimiento varchar2(15) not null,
  8  Calle varchar2(10),
  9  Num_casa varchar2(5) not  null,
 10  Cod_cargo varchar2(5),
 11  Constraint fk_cod_cargo foreign key (Cod_cargo) references ti
po_personal(Cod_cargo)
 12  );

Table created.
```

-- **Creación de tabla teléfono personal**

Create table telefono_personal(
Dni varchar2(15) ,

Constraint fk_dni foreign key (DNI) references Personal(DNI),
Cod_telf varchar2(5) ,
Constraint fk_cod_telf_t foreign key(cod_telf) references tipo_telefono(cod_telf),
Constraint telefono_personal primary key (dni,cod_telf)
);

```
SQL> Create table telefono_personal(
  2  Dni varchar2(15) ,
  3  Constraint fk_dni foreign key (DNI) references Personal(DNI),

  4  Cod_telf varchar2(5) ,
  5  Constraint fk_cod_telf_t foreign key(cod_telf) references tip
o_telefono(cod_telf),
  6  Constraint telefono_personal primary key (dni,cod_telf)
  7  );

Table created.
```

ALTER TABLE TELEFONO_PERSONAL ADD TELEFONO NUMBER;

```
SQL> ALTER TABLE TELEFONO_PERSONAL ADD TELEFONO NUMBER;

Table altered.
```

-- **Creación de tabla Repartidor**

Create table Repartidor(
Licencia varchar2(5) not null primary key,
DNI varchar2(15),
Constraint fk_dni_R foreign key (DNI) references Personal(DNI),
Matricula varchar2(10),
Constraint fk_matricula foreign key(matricula) references transporte(matricula)
);

```
SQL> Create table Repartidor(
  2  Licencia varchar2(5) not null primary key,
  3  DNI varchar2(15),
  4  Constraint fk_dni_R foreign key (DNI) references Personal(DNI
),
  5  Matricula varchar2(10),
  6  Constraint fk_matricula foreign key(matricula) references tra
nsporte(matricula)
  7  );

Table created.
```

-- **Creación de tabla Donador_Insumo**

Create table Donador_insumo (
Dni_Ruc varchar2(15),
Constraint fk_dni_D foreign key (DNI_ruc) references Donador(Dni_ruc),
Cod_donacion number,

Constraint fk_cod_donacion foreign key (cod_donacion) references Insumo(cod_donacion
),
Constraint Donador_insumo primary key (dni_ruc,cod_donacion)
);

```
SQL> Create table Donador_insumo (
  2  Dni_Ruc varchar2(15),
  3  Constraint fk_dni_D foreign key (DNI_ruc) references Donac
Dni_ruc),
  4  Cod_donacion number,
  5  Constraint fk_cod_donacion foreign key (cod_donacion) refe
ces Insumo(cod_donacion),
  6  Constraint Donador_insumo primary key (dni_ruc,cod_donacio
  7  );

Table created.
```

ALTER TABLE DONADOR_INSUMO ADD COD_INSUMO VARCHAR2(5);

```
SQL> ALTER TABLE DONADOR_INSUMO ADD COD_INSUMO VARCHAR2(5);

Table altered.
```

ALTER TABLE DONADOR_INSUMO ADD constraint fk_a_cod_insumo foreign key (Cod_insumo) references Tipo_insumo(cod_insumo);

```
SQL> ALTER TABLE DONADOR_INSUMO ADD  constraint fk_a_cod_insumo forei
gn key (Cod_insumo) references Tipo_insumo(cod_insumo);

Table altered.
```

ALTER TABLE Donador_insumo add cantidad number;

```
SQL> ALTER TABLE Donador_insumo add cantidad number;

Table altered.
```

-- **Creación de tabla Insumo_Personal_Instalación_medica**

Create table Insumo_personal_IM(
Cod_insumo varchar2(5),
Constraint fk_cod_insumo_I foreign key (cod_insumo) references tipo_Insumo(cod_insum
o),
DNI varchar2(15),
Constraint fk_dni_I foreign key (DNI) references Personal(DNI),

Codigo_IM varchar2(5) not null,
Constraint fk_codigo_im_I foreign key (codigo_IM) references Instalaciones_medicas(Codigo_IM),
Constraint Insumo_personal_IM primary key (Cod_insumo,dni,codigo_im),
Fecha_dist date,
Cantidad_llevada number
);

```
SQL> Create table Insumo_personal_IM(
  2  Cod_insumo varchar2(5),
  3  Constraint fk_cod_insumo_I foreign key (cod_insumo) reference
s tipo_Insumo(cod_insumo),
  4  DNI varchar2(15),
  5  Constraint fk_dni_I foreign key (DNI) references Personal(DNI
),
  6  Codigo_IM varchar2(5) not null,
  7  Constraint fk_codigo_im_I foreign key (codigo_IM) references
Instalaciones_medicas(Codigo_IM),
  8  Constraint Insumo_personal_IM primary key (Cod_insumo,dni,cod
igo_im),
  9  Fecha_dist date,
 10  Cantidad_llevada number
 11  );

Table created.
```

-- **Creación de tabla Donador_Cedis**

Create table Donador_CEDIS(
Dni_Ruc varchar2(15),
Constraint fk_dni_CE foreign key (DNI_ruc) references Donador(Dni_ruc),
Codigo_cedis varchar2(5),
Constraint fk_codigo_cedis foreign key (codigo_cedis) references centro_distribucion(codigo_cedis),
Constraint Donador_CEDIS primary key (DNI_RUC,codigo_cedis)
);

```
SQL> Create table Donador_CEDIS(
  2  Dni_Ruc varchar2(15),
  3  Constraint fk_dni_CE foreign key (DNI_ruc) references Donador
(Dni_ruc),
  4  Codigo_cedis varchar2(5),
  5  Constraint fk_codigo_cedis foreign key (codigo_cedis) referen
ces centro_distribucion(codigo_cedis),
  6  Constraint Donador_CEDIS primary key (DNI_RUC,codigo_cedis)
  7  );

Table created.
```

-- **Creación de tabla Personal Cedis**

Create table personal_CEDIS(
DNI varchar2(15),
Constraint fk_dni_P foreign key (DNI) references Personal(DNI),

Codigo_cedis varchar2(5),
Constraint fk_codigo_cedis_P foreign key (codigo_cedis) references centro_distribucion(codigo_cedis),
Constraint personal_CEDIS primary key (DNI,codigo_cedis)
);

```
SQL> Create table personal_CEDIS(
  2  DNI varchar2(15),
  3  Constraint fk_dni_P foreign key (DNI) referenc
),
  4  Codigo_cedis varchar2(5),
  5  Constraint fk_codigo_cedis_P foreign key (codi
ences centro_distribucion(codigo_cedis),
  6  Constraint personal_CEDIS primary key (DNI,coc
  7  );

Table created.
```

------CREACIÓN DE TABLA PARA REGISTRAR STATUS DEL PERSONAL

Create table registro_personal(
DNI varchar2(15) not null primary key,
Status varchar2(20) not null,
Fecha date
);

```
SQL> Create table registro_personal(
  2  DNI varchar2(15) not null primary key,
  3  Status varchar2(20) not null,
  4  Fecha date
  5  );

Table created.
```

----CREACION DE TABLA PARA REGISTRAR TRANSPORTES NUEVOS O DAÑADOS

Create table registro_Transporte(
Matricula varchar2(15) not null primary key,
Cap_carga number not null,
Cod_trans varchar2(5),
Status varchar2(20) not null,
Fecha date
);

```
SQL> Create table registro_Transporte(
  2  Matricula varchar2(15) not null primary key,
  3  Cap_carga number not null,
  4  Cod_trans varchar2(5),
  5  Status varchar2(20) not null,
  6  Fecha date
  7  );

Table created.
```

PROCEDIMIENTOS UTILIZADOS

--PROCEDIMIENTO PARA INSERTAR EN TIPO TELEFONO

```
Create or Replace Procedure P_Tipo_tel(
  P_cod_telf Tipo_Telefono.Cod_Telf%TYPE,
  p_desc_telf Tipo_Telefono.Desc_Telf%TYPE
) AS
BEGIN
INSERT INTO Tipo_Telefono (Cod_Telf, Desc_Telf) VALUES (  P_cod_telf,  p_desc_telf);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Tipo_tel;
/
```

```
SQL> Create or Replace Procedure P_Tipo_tel(
  2    P_cod_telf Tipo_Telefono.Cod_Telf%TYPE,
  3    p_desc_telf Tipo_Telefono.Desc_Telf%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Tipo_Telefono (Cod_Telf, Desc_Telf) VALUES (  P_c
od_telf,  p_desc_telf);
  7  Exception
  8  When others then
  9  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 10  COMMIT;
 11  END P_Tipo_tel;
 12  /

Procedure created.
```

--PROCEDIMIENTO PARA INSERTAR EN TIPO CORREOS

```
CREATE OR REPLACE PROCEDURE P_Tipos_Correos(
p_Cod_Correo          Tipo_Email.Cod_Email%TYPE,
p_Desc_Correo         Tipo_Email.Desc_Email%TYPE
) AS
BEGIN
INSERT INTO Tipo_Email (Cod_Email, Desc_Email) VALUES
(  p_Cod_Correo,  p_Desc_Correo);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Tipos_Correos;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE P_Tipos_Correos(
  2  p_Cod_Correo           Tipo_Email.Cod_Email%TYPE,
  3  p_Desc_Correo          Tipo_Email.Desc_Email%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Tipo_Email (Cod_Email, Desc_Email) VALUES (  p_Co
d_Correo,  p_Desc_Correo);
  7  Exception
  8  When others then
  9  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 10  COMMIT;
 11  END P_Tipos_Correos;
 12  /

Procedure created.
```

--PROCEDIMIENTO PARA INSERTAR EN TIPO INSUMO

```
Create or Replace Procedure P_Tipo_Insum(
 P_cod_insumo     Tipo_Insumo.cod_insumo%TYPE,
 p_desc_insum      Tipo_Insumo.desc_insumo%TYPE
) AS
BEGIN
INSERT INTO Tipo_Insumo(cod_insumo,desc_insumo) VALUES
(P_cod_insumo, p_desc_insum);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Tipo_Insum;
/
```

```
SQL> Create or Replace Procedure P_Tipo_Insum(
  2   P_cod_insumo     Tipo_Insumo.cod_insumo%TYPE,
  3   p_desc_insum      Tipo_Insumo.desc_insumo%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Tipo_Insumo(cod_insumo,desc_insumo) VALUES (P_cod
_insumo, p_desc_insum);
  7  Exception
  8  When others then
  9  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 10  COMMIT;
 11  END P_Tipo_Insum;
 12  /

Procedure created.
```

--PROCEDIMIENTO PARA INSERTAR EN TIPO TRANSPORTE

```
Create or Replace Procedure P_Tipo_Transp(
 P_cod_trans      Tipo_transporte.cod_trans%TYPE,
 p_desc_trans     Tipo_transporte.desc_trans%TYPE
) AS
BEGIN
INSERT INTO Tipo_transporte(cod_trans,desc_trans) VALUES
(P_cod_trans, p_desc_trans);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Tipo_Transp;
/
```

```
SQL> Create or Replace Procedure P_Tipo_Transp(
  2   P_cod_trans      Tipo_transporte.cod_trans%TYPE,
  3   p_desc_trans     Tipo_transporte.desc_trans%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Tipo_transporte(cod_trans,desc_trans) VALUES (P_c
od_trans, p_desc_trans);
  7  Exception
  8  When others then
  9  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 10  COMMIT;
 11  END P_Tipo_Transp;
 12  /

Procedure created.
```

--PROCEDIMIENTO PARA INSERTAR EN TIPO PERSONAL

```
Create or Replace Procedure P_Tipo_Person(
 P_cod_cargo     Tipo_personal.cod_cargo%TYPE,
 p_descripcion      Tipo_personal.descripcion%TYPE
    ) AS
BEGIN
INSERT INTO Tipo_personal(cod_cargo,descripcion) VALUES (P_cod_cargo, p_descripcion);
```

Exception

When others then

DBMS_OUTPUT.PUT_LINE('REGISTRO NO CREADO');

COMMIT;

END P_Tipo_Person;

/

```
SQL> Create or Replace Procedure P_Tipo_Person(
  2   P_cod_cargo     Tipo_personal.cod_cargo%TYPE,
  3   p_descripcion      Tipo_personal.descripcion%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Tipo_personal(cod_cargo,descripcion) VALUES (P_co
d_cargo, p_descripcion);
  7  Exception
  8  When others then
  9  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 10  COMMIT;
 11  END P_Tipo_Person;
 12  /

Procedure created.
```

--PROCEDIMIENTO PARA INSERTAR EN EMAIL DEL DONADOR

```
CREATE OR REPLACE PROCEDURE P_Email_donad(
P_email          Email_donador.email%TYPE,
p_dni_ruc         Email_donador.dni_ruc%TYPE,
p_cod_email          Email_donador.Cod_Email%TYPE
) AS
BEGIN
INSERT INTO Email_donador (email,dni_ruc,cod_email) VALUES
(P_email,P_dni_ruc,P_cod_email);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Email_donad;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE P_Email_donad(
  2  P_email             Email_donador.email%TYPE,
  3  p_dni_ruc            Email_donador.dni_ruc%TYPE,
  4  p_cod_email             Email_donador.Cod_Email%TYPE
  5  ) AS
  6  BEGIN
  7  INSERT INTO Email_donador (email,dni_ruc,cod_email) VALUES (P
_email,P_dni_ruc,P_cod_email);
  8  Exception
  9  When others then
 10  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 11  COMMIT;
 12  END P_Email_donad;
 13  /

Procedure created.
```

--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA DONADOR

```
CREATE OR REPLACE PROCEDURE p_DONADOR (
P_dni_ruc          DONADOR.dni_ruc%Type,
p_NOMBRE     DONADOR.nombre%TYPE,
p_APELLIDO     DONADOR.apellido%TYPE,
p_PROVINCIA    DONADOR.provincia%TYPE,
p_DISTRITO    DONADOR.distrito%TYPE,
p_CORREGIMIENTO    DONADOR.corregimiento%TYPE,
p_CALLE DONADOR.calle%TYPE,
p_NUM_CASA DONADOR.no_casa%TYPE
)
AS
BEGIN
INSERT INTO DONADOR ( dni_ruc,NOMBRE, APELLIDO, PROVINCIA, DISTRITO,
CORREGIMIEnTO, CALLE, NO_CASA)
Values (p_dni_ruc, p_NOMBRE, p_APELLIDO, p_PROVINCIA, p_DISTRITO,
p_CORREGIMIENTO, p_CALLE, p_NUM_CASA);
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE ('REGISTRO NO CREADO');
COMMIT;
END p_DONADOR;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE p_DONADOR (
  2  P_dni_ruc             DONADOR.dni_ruc%Type,
  3  p_NOMBRE      DONADOR.nombre%TYPE,
  4  p_APELLIDO      DONADOR.apellido%TYPE,
  5  p_PROVINCIA    DONADOR.provincia%TYPE,
  6  p_DISTRITO    DONADOR.distrito%TYPE,
  7  p_CORREGIMIENTO    DONADOR.corregimiento%TYPE,
  8  p_CALLE DONADOR.calle%TYPE,
  9  p_NUM_CASA DONADOR.no_casa%TYPE
 10  )
 11  AS
 12  BEGIN
 13  INSERT INTO DONADOR ( dni_ruc,NOMBRE, APELLIDO, PROVINCIA, DI
STRITO, CORREGIMIEnTO, CALLE, NO_CASA)
 14  Values (p_dni_ruc, p_NOMBRE, p_APELLIDO, p_PROVINCIA, p_DISTR
ITO, p_CORREGIMIENTO, p_CALLE, p_NUM_CASA);
 15  EXCEPTION
 16  WHEN OTHERS THEN
 17  DBMS_OUTPUT.PUT_LINE ('REGISTRO NO CREADO');
 18  COMMIT;
 19  END p_DONADOR;
 20  /

Procedure created.
```

--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA TELEFONO_DONADOR

CREATE OR REPLACE PROCEDURE P_Telefono_donador(

p_DNI_RUC Telefono_donador.DNI_RUC%TYPE,

p_Cod_Telf Telefono_donador.Cod_Telf%TYPE,

p_telefono Telefono_donador.telefono%TYPE

) AS

BEGIN

INSERT INTO Telefono_Donador (DNI_RUC, Cod_Telf,TELEFONO) VALUES (p_DNI_RUC, p_Cod_Telf,P_TELEFONO);

Exception

When others then

DBMS_OUTPUT.PUT_LINE('REGISTRO NO CREADO');

COMMIT;

END P_Telefono_donador;

/

```
SQL> CREATE OR REPLACE PROCEDURE P_Telefono_donador(
  2  p_DNI_RUC  Telefono_donador.DNI_RUC%TYPE,
  3  p_Cod_Telf  Telefono_donador.Cod_Telf%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Telefono_Donador (DNI_RUC, Cod_Telf ) VALUES (p_DNI_
RUC, p_Cod_Telf);
  7  Exception
  8  When others then
  9  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 10  COMMIT;
 11  END P_Telefono_donador;
 12  /

Procedure created.
```

---PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA TELEFONO_PERSONAL

```
Create or Replace Procedure P_telf_Personal(
  P_DNI  telefono_personal.Dni%TYPE,
  P_cod_telf Telefono_personal.Cod_Telf%TYPE,
  p_Telefono Telefono_personal.telefono%TYPE
   ) AS
  BEGIN
INSERT INTO Telefono_personal (Dni,Cod_Telf, telefono)
 VALUES (  p_DNI,P_cod_telf,  p_Telefono);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Telf_Personal;
/
```

```
SQL> Create or Replace Procedure P_telf_Personal(
  2    P_DNI  telefono_personal.Dni%TYPE,
  3    P_cod_telf Telefono_personal.Cod_Telf%TYPE,
  4    p_Telefono Telefono_personal.telefono%TYPE
  5     ) AS
  6    BEGIN
  7  INSERT INTO Telefono_personal (Dni,Cod_Telf, telefono)
  8   VALUES (  p_DNI,P_cod_telf,  p_Telefono);
  9  Exception
 10  When others then
 11  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 12  COMMIT;
 13  END P_Telf_Personal;
 14  /

Procedure created.
```

---PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA TELEFONO_INSTALACION_MEDICA

```
CREATE OR REPLACE PROCEDURE P_Telefono_IM(
p_Codigo_IM  telefono_im.Codigo_IM%TYPE,
p_Cod_telf  telefono_im.Cod_telf%TYPE,
P_TELEFONO TELEFONO_IM.TELEFONO%TYPE
) AS
BEGIN
INSERT INTO telefono_im (Codigo_IM, Cod_telf,TELEFONO) VALUES (p_Codigo_IM,
p_Cod_telf,P_TELEFONO);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Telefono_IM;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE P_Telefono_IM(
  2  p_Codigo_IM  telefono_im.Codigo_IM%TYPE,
  3  p_Cod_telf  telefono_im.Cod_telf%TYPE,
  4  P_TELEFONO TELEFONO_IM.TELEFONO%TYPE
  5  ) AS
  6  BEGIN
  7  INSERT INTO telefono_im (Codigo_IM, Cod_telf,TELEFONO) VALUES
 (p_Codigo_IM, p_Cod_telf,P_TELEFONO);
  8  Exception
  9  When others then
 10  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 11  COMMIT;
 12  END P_Telefono_IM;
 13  /

Procedure created.
```

------**PROCEDIMIENTO DE INSUMO_PERSONAL_IM**

Create or replace procedure P_distribucion(

p_cod_insumo insumo_personal_im.cod_insumo%TYPE,

p_DNI insumo_personal_im.DNI%TYPE,

p_codigo_im insumo_personal_im.codigo_im %TYPE,

p_fecha_dist insumo_personal_im.fecha_dist%TYPE,

p_cantidad_llevada insumo_personal_im.cantidad_llevada%TYPE

)as

BEGIN

Insert into insumo_personal_im values(p_cod_insumo, p_DNI, p_codigo_im, p_fecha_dist, p_cantidad_llevada);

Exception

When others then

DBMS_OUTPUT.PUT_LINE('REGISTRO NO CREADO');

COMMIT;

END P_distribucion;

/

```
SQL> Create or replace procedure P_distribucion(
  2  p_cod_insumo insumo_personal_im.cod_insumo%TYPE,
  3  p_DNI insumo_personal_im.DNI%TYPE,
  4  p_codigo_im insumo_personal_im.codigo_im %TYPE,
  5  p_fecha_dist insumo_personal_im.fecha_dist%TYPE,
  6  p_cantidad_llevada insumo_personal_im.cantidad_llevada%TYPE
  7  )as
  8  BEGIN
  9  Insert into insumo_personal_im values(p_cod_insumo, p_DNI, p_
codigo_im, p_fecha_dist, p_cantidad_llevada);
 10  Exception
 11  When others then
 12  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 13  COMMIT;
 14  END P_distribucion;
 15  /

Procedure created.
```

--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA DONADOR_INSUMO

```
CREATE OR REPLACE PROCEDURE P_Donador_insumo (

p_Dni_Ruc  Donador_insumo.DNI_RUC%TYPE,

p_Cod_donacion  Donador_insumo.Cod_donacion%TYPE,

P_COD_INSUMO DONADOR_INSUMO.COD_INSUMO%TYPE,

p_cantidad Donador_insumo.cantidad%TYPE

) AS

BEGIN

INSERT INTO Donador_insumo (Dni_Ruc,Cod_donacion, COD_INSUMO,cantidad) VALUES
(p_Dni_Ruc, p_Cod_donacion, P_COD_INSUMO,p_cantidad);

Exception

When others then

DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');

COMMIT;

END P_Donador_insumo;

/
```

```
SQL> CREATE OR REPLACE PROCEDURE P_Donador_insumo (
  2  p_Dni_Ruc  Donador_insumo.DNI_RUC%TYPE,
  3  p_Cod_donacion  Donador_insumo.Cod_donacion%TYPE,
  4  P_COD_INSUMO DONADOR_INSUMO.COD_INSUMO%TYPE,
  5  p_cantidad Donador_insumo.cantidad%TYPE
  6  ) AS
  7  BEGIN
  8  INSERT INTO Donador_insumo (Dni_Ruc,Cod_donacion, COD_INSUMO,c
antidad) VALUES (p_Dni_Ruc, p_Cod_donacion, P_COD_INSUMO,p_cantidad
);
  9  Exception
 10  When others then
 11  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 12  COMMIT;
 13  END P_Donador_insumo;
 14  /

Procedure created.
```

--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA CENTRO_DISTRIBUCION

```
CREATE OR REPLACE PROCEDURE p_Centro_distr(
P_cod_cedis          Centro_distribucion.codigo_cedis%TYPE,
p_nombre     Centro_distribucion.nombre%Type,
p_Region     Centro_distribucion.Region%Type,
p_Provincia    Centro_distribucion.Provincia%Type,
p_Ubicacion    Centro_distribucion.ubicacion %Type
 )
 AS
BEGIN
INSERT INTO Centro_distribucion (Codigo_cedis,Nombre,Region,Provincia, Ubicacion)
Values (p_cod_cedis, p_nombre, p_Region, p_Provincia, p_Ubicacion);
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE ('REGISTRO NO CREADO');
COMMIT;
END p_Centro_distr;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE p_Centro_distr(
  2  P_cod_cedis             Centro_distribucion.codigo_cedis%TYPE,
  3  p_nombre      Centro_distribucion.nombre%Type,
  4  p_Region      Centro_distribucion.Region%Type, p_Provincia    Centro_distribucion.Provincia%Type,
  5  p_Ubicacion    Centro_distribucion.ubicacion %Type
  6    )
  7    AS
  8  BEGIN
  9  INSERT INTO Centro_distribucion (Codigo_cedis,Nombre,Region,Provincia, Ubicacion)
 10  Values (p_cod_cedis, p_nombre, p_Region, p_Provincia, p_Ubicacion);
 11  EXCEPTION
 12  WHEN OTHERS THEN
 13  DBMS_OUTPUT.PUT_LINE ('REGISTRO NO CREADO');
 14  COMMIT;
 15  END p_Centro_distr;
 16  /

Procedure created.
```

---PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA INSTALACIONES MEDICAS

```
 CREATE OR REPLACE PROCEDURE p_instalaciones (
P_cod_IM         Instalaciones_medicas.codigo_IM%TYPE,
p_nombre     Instalaciones_medicas.nombre%TYPE,
p_Region     Instalaciones_medicas.Region%TYPE,
p_Ubicacion    Instalaciones_medicas.ubicacion%Type
 )
 AS
```

```
BEGIN
INSERT INTO Instalaciones_medicas (Codigo_IM, Nombre,Region, Ubicacion)
Values (p_cod_IM, p_nombre, p_Region, p_Ubicacion);
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE ('REGISTRO NO CREADO');
COMMIT;
END p_instalaciones;
/
```

```
SQL>  CREATE OR REPLACE PROCEDURE p_instalaciones (
  2  P_cod_IM           Instalaciones_medicas.codigo_IM%TYPE,
  3  p_nombre      Instalaciones_medicas.nombre%TYPE,
  4  p_Region      Instalaciones_medicas.Region%TYPE,
  5  p_Ubicacion    Instalaciones_medicas.ubicacion%Type
  6    )
  7    AS
  8  BEGIN
  9  INSERT INTO Instalaciones_medicas (Codigo_IM, Nombre,Region, Ubicacion)
 10  Values (p_cod_IM, p_nombre, p_Region, p_Ubicacion);
 11  EXCEPTION
 12  WHEN OTHERS THEN
 13  DBMS_OUTPUT.PUT_LINE ('REGISTRO NO CREADO');
 14  COMMIT;
 15  END p_instalaciones;
 16  /

Procedure created.
```

--PROCEDIMIENTO PARA CARGAR DATOS EN LA TABLA DONADOR_CEDIS

```
Create or replace procedure P_donador_cedis(
P_Dni_Ruc Donador_CEDIS.Dni_Ruc%TYPE,
P_Codigo_cedis Donador_CEDIS.Codigo_cedis%TYPE
)AS
BEGIN
Insert into Donador_CEDIS(Dni_Ruc, Codigo_cedis)
Values(p_Dni_Ruc,p_Codigo_cedis);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_donador_cedis;
/
```

```
SQL> Create or replace procedure P_donador_cedis(
  2  P_Dni_Ruc Donador_CEDIS.Dni_Ruc%TYPE,
  3  P_Codigo_cedis Donador_CEDIS.Codigo_cedis%TYPE
  4  )AS
  5  BEGIN
  6  Insert into Donador_CEDIS(Dni_Ruc, Codigo_cedis)
  7  Values(p_Dni_Ruc,p_Codigo_cedis);
  8  Exception
  9  When others then
 10  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 11  COMMIT;
 12  END P_donador_cedis;
 13  /

Procedure created.
```

--PROCEDIMIENTO PARA CARGAR DATOS EN LA TABLA PERSONAL_CEDIS

```
Create or replace procedure P_personal_cedis(
P_Dni personal_CEDIS.Dni%TYPE,
P_Codigo_cedis personal_CEDIS.Codigo_cedis%TYPE
)AS
BEGIN
Insert into personal_CEDIS(Dni, Codigo_cedis)
Values(p_Dni, p_Codigo_cedis);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_personal_cedis;
/
```

```
SQL> Create or replace procedure P_personal_cedis(
  2  P_Dni personal_CEDIS.Dni%TYPE,
  3  P_Codigo_cedis personal_CEDIS.Codigo_cedis%TYPE
  4  )AS
  5  BEGIN
  6  Insert into personal_CEDIS(Dni, Codigo_cedis)
  7  Values(p_Dni, p_Codigo_cedis);
  8  Exception
  9  When others then
 10  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 11  COMMIT;
 12  END P_personal_cedis;
 13  /

Procedure created.
```

--PROCEDIMIENTO PARA CARGAR DATOS EN LA TABLA TRANSPORTE

```
Create or replace procedure P_transporte(
P_Matricula transporte.matricula%TYPE,
P_Cap_carga transporte.cap_carga%TYPE,
P_Cod_trans transporte.cod_trans%TYPE
)AS
BEGIN
Insert into transporte(matricula, cap_carga, cod_trans)
Values(p_matricula, p_cap_carga, P_cod_trans);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_transporte;
/
```

```
SQL> Create or replace procedure P_transporte(
  2  P_Matricula transporte.matricula%TYPE,
  3  P_Cap_carga transporte.cap_carga%TYPE,
  4  P_Cod_trans transporte.cod_trans%TYPE
  5  )AS
  6  BEGIN
  7  Insert into transporte(matricula, cap_carga, cod_trans)
  8  Values(p_matricula, p_cap_carga, P_cod_trans);
  9  Exception
 10  When others then
 11  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 12  COMMIT;
 13  END P_transporte;
 14  /

Procedure created.
```

--PROCEDIMIENTO PARA CARGAR DATOS A LA TABLA PERSONAL

CREATE OR REPLACE PROCEDURE P_Personal (

p_Dni Personal.DNI%TYPE,

p_Nombre Personal.Nombre%TYPE,

p_Apellido Personal.Apellido%TYPE,

p_Fecha_nac Personal.fecha_nac%TYPE,

p_Ciudad Personal.ciudad%TYPE,

p_Corregimiento Personal.corregimiento%TYPE,

p_calle Personal.calle%TYPE,

p_num_casa Personal.num_casa%TYPE,

p_cod_cargo Personal.cod_cargo%TYPE

) AS

BEGIN

INSERT INTO Personal (Dni,Nombre, Apellido, Fecha_nac, Ciudad, Corregimiento, Calle, num_casa, cod_cargo) VALUES (p_Dni,p_Nombre, p_Apellido, p_Fecha_nac, p_Ciudad, p_Corregimiento, p_Calle, p_num_casa, p_cod_cargo);

Exception

When others then

DBMS_OUTPUT.PUT_LINE('REGISTRO NO CREADO');

COMMIT;

END P_personal;

```
SQL> --PROCEDIMIENTO PARA CARGAR DATOS A LA TABLA PERSONAL
SQL> CREATE OR REPLACE PROCEDURE P_Personal (
  2  p_Dni    Personal.DNI%TYPE,
  3  p_Nombre  Personal.Nombre%TYPE,
  4  p_Apellido  Personal.Apellido%TYPE,
  5  p_Fecha_nac  Personal.fecha_nac%TYPE,
  6  p_Ciudad  Personal.ciudad%TYPE,
  7  p_Corregimiento  Personal.corregimiento%TYPE,
  8  p_calle  Personal.calle%TYPE,
  9  p_num_casa  Personal.num_casa%TYPE,
 10  p_cod_cargo  Personal.cod_cargo%TYPE
 11   ) AS
 12  BEGIN
 13  INSERT INTO Personal  (Dni,Nombre, Apellido, Fecha_nac, Ciudad,
Corregimiento, Calle, num_casa, cod_cargo) VALUES (p_Dni,p_Nombre, p_
Apellido, p_Fecha_nac, p_Ciudad, p_Corregimiento,      p_Calle, p_num
_casa, p_cod_cargo);
 14  Exception
 15  When others then
 16  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 17  COMMIT;
 18  END P_personal;
 19  /

Procedure created.
```

/

--PROCEDIMIENTO PARA CARGAR DATOS EN LA TABLA INSUMO

CREATE OR REPLACE PROCEDURE P_Insumo (

p_Cod_donacion Insumo.cod_donacion%TYPE,

p_Cod_insumo Insumo.Cod_insumo%TYPE,

p_cantidad Insumo.cantidad_TOTAL%TYPE,

P_Descripcion Insumo.descripcion%type

) AS

BEGIN

INSERT INTO insumo (Cod_donacion, Cod_insumo, cantidad_TOTAL, descripcion) VALUES (P_Cod_donacion, p_Cod_insumo, p_cantidad,p_descripcion);

Exception

When others then

DBMS_OUTPUT.PUT_LINE('REGISTRO NO CREADO');

COMMIT;

END P_insumo;

/

```
SQL> CREATE OR REPLACE PROCEDURE P_Insumo (
  2  p_Cod_donacion  Insumo.cod_donacion%TYPE,
  3  p_Cod_insumo  Insumo.Cod_insumo%TYPE,
  4  p_cantidad Insumo.cantidad_TOTAL%TYPE,
  5  P_Descripcion Insumo.descripcion%type
  6  ) AS
  7  BEGIN
  8  INSERT INTO insumo (Cod_donacion,  Cod_insumo, cantidad_TOTAL,
 descripcion) VALUES (P_Cod_donacion,  p_Cod_insumo, p_cantidad,p_d
escripcion);
  9  Exception
 10  When others then
 11  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 12  COMMIT;
 13  END P_insumo;
 14  /
```

----PROCEDIMIENTO PARA CARGAR DATOS EN LA TABLA REPARTIDOR

CREATE OR REPLACE PROCEDURE P_Repartidor(
v_Licencia Repartidor.Licencia%TYPE,
v_DNI Repartidor.DNI%TYPE,
v_Matricula Repartidor.Matricula%TYPE
)
AS
BEGIN
INSERT INTO Repartidor (Licencia, DNI, Matricula) VALUES (v_Licencia, v_DNI,
v_Matricula);
Exception
When others then
DBMS_OUTPUT.PUT_LINE('REGISTRO NO CREADO');
COMMIT;
END P_Repartidor;
/

```
SQL> CREATE OR REPLACE PROCEDURE P_Repartidor(
  2  v_Licencia Repartidor.Licencia%TYPE,
  3  v_DNI  Repartidor.DNI%TYPE,
  4  v_Matricula  Repartidor.Matricula%TYPE
  5  )
  6  AS
  7  BEGIN
  8  INSERT INTO  Repartidor (Licencia, DNI, Matricula ) VALUES (v
_Licencia, v_DNI, v_Matricula);
  9  Exception
 10  When others then
 11  DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
 12  COMMIT;
 13  END P_Repartidor;
 14  /

Procedure created.
```

----FUNCION PARA PROCEDIMIENTO TABLA INSUMO_INSTALACION MEDICA Y PERSONAL

```
Create or replace function f_cal_cantidad(

f_cantidad  insumo.CANTIDAD_TOTAL%TYPE,

f_cantidad_llevada  Insumo_personal_IM.cantidad_llevada%TYPE

)return number

AS

Cantidad_nueva number;

BEGIN

Cantidad_nueva:=f_cantidad-f_cantidad_llevada;

RETURN Cantidad_nueva;

END f_cal_cantidad;

/
```

```
SQL> Create or replace function f_cal_cantidad(
  2  f_cantidad  insumo.CANTIDAD_TOTAL%TYPE,
  3  f_cantidad_llevada  Insumo_personal_IM.cantidad_llevada%TYPE
  4  )return number
  5  AS
  6  Cantidad_nueva number;
  7  BEGIN
  8  Cantidad_nueva:=f_cantidad-f_cantidad_llevada;
  9  RETURN Cantidad_nueva;
 10  END f_cal_cantidad;
 11  /

Function created.
```

--PROCEDIMIENTO PARA CARGAR Y ACTUALIZAR TABLA INSUMO_INSTALACION MEDICA Y PERSONAL

Create or replace procedure P_cantidad_distribuida

AS

p_cod_insumo insumo.cod_insumo%TYPE;

p_id_insumo Insumo_personal_IM.cod_insumo%TYPE;

p_cantidad insumo.cantidad_total%TYPE;

p_cantidad_llevada Insumo_personal_IM.cantidad_llevada%TYPE;

p_descripcion insumo.descripcion%TYPE;

CURSOR recorre_insumo is

Select cod_insumo, cantidad_total, descripcion from insumo;

BEGIN

OPEN recorre_insumo;

LOOP

FETCH recorre_insumo INTO p_cod_insumo, p_cantidad, p_descripcion;

--Select cantidad_total into p_cantidad from insumo where cod_insumo=p_cod_insumo;

/*Select cantidad_llevada into p_cantidad_llevada from Insumo_personal_IM where cod_insumo=p_id_insumo;*/

IF (p_cantidad_llevada>p_cantidad) THEN

DBMS_OUTPUT.PUT_LINE('Insumo agotado');

ELSIF (p_cantidad_llevada<=p_cantidad) THEN

UPDATE INSUMO

Set

Cantidad_total= f_cal_cantidad(p_cantidad,p_cantidad_llevada)

Where p_cod_insumo=p_id_insumo;

ELSE

DBMS_OUTPUT.PUT_LINE('Insumo no encontrado');

END IF;

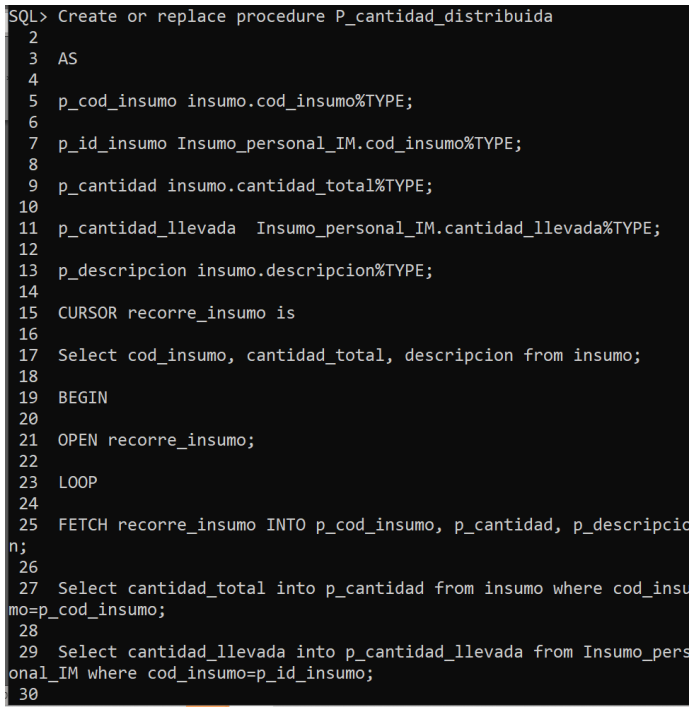
EXIT WHEN recorre_insumo%NOTFOUND;

END LOOP;

CLOSE recorre_insumo;

END P_cantidad_distribuida;

/



```
SQL> Create or replace procedure P_cantidad_distribuida
  2
  3  AS
  4
  5  p_cod_insumo insumo.cod_insumo%TYPE;
  6
  7  p_id_insumo Insumo_personal_IM.cod_insumo%TYPE;
  8
  9  p_cantidad insumo.cantidad_total%TYPE;
 10
 11  p_cantidad_llevada  Insumo_personal_IM.cantidad_llevada%TYPE;
 12
 13  p_descripcion insumo.descripcion%TYPE;
 14
 15  CURSOR recorre_insumo is
 16
 17  Select cod_insumo, cantidad_total, descripcion from insumo;
 18
 19  BEGIN
 20
 21  OPEN recorre_insumo;
 22
 23  LOOP
 24
 25  FETCH recorre_insumo INTO p_cod_insumo, p_cantidad, p_descripcio
n;
 26
 27  Select cantidad_total into p_cantidad from insumo where cod_insu
mo=p_cod_insumo;
 28
 29  Select cantidad_llevada into p_cantidad_llevada from Insumo_pers
onal_IM where cod_insumo=p_id_insumo;
 30
```

```
31  IF (p_cantidad_llevada>p_cantidad) THEN
32
33  DBMS_OUTPUT.PUT_LINE('Insumo agotado');
34
35  ELSIF (p_cantidad_llevada<=p_cantidad) THEN
36
37  UPDATE INSUMO
38
39  Set
40
41  Cantidad_total= f_cal_cantidad(p_cantidad,p_cantidad_llevada)
42
43  Where p_cod_insumo=p_id_insumo;
44
45  ELSE
46
47  DBMS_OUTPUT.PUT_LINE('Insumo no encontrado');
48
49  END IF;
50
51  EXIT WHEN recorre_insumo%NOTFOUND;
52
53  END LOOP;
54
55  CLOSE  recorre_insumo;
56
57  END P_cantidad_distribuida;
58
59  /

Procedure created.
```

IMPLEMENTACION DE TRIGGERS

--TRIGGER PARA INSERTAR Y ELIMINAR REGISTROS DE LA TABLA PERSONAL

Create or replace trigger T_personal

After insert or delete on Personal

For each row

BEGIN

IF INSERTING THEN

INSERT INTO registro_personal (DNI, Status, Fecha)

  VALUES (:new.DNI, ‘INSERTADO’, SYSDATE);

ELSIF DELETING THEN

INSERT INTO registro_personal (DNI, Status, Fecha)

  VALUES (:old.DNI, ‘ELIMINADO’, SYSDATE);

END IF;

END T_personal;

/

```
SQL> Create or replace trigger T_personal
  2  After insert or delete  on Personal
  3  For each row
  4
  5  BEGIN
  6
  7  IF INSERTING THEN
  8  INSERT INTO registro_personal ( DNI, Status, Fecha)
  9    VALUES (:new.DNI, 'INSERTADO', SYSDATE);
 10
 11  ELSIF DELETING THEN
 12  INSERT INTO registro_personal ( DNI, Status, Fecha)
 13    VALUES (:OLD.DNI, 'ELIMINADO', SYSDATE);
 14
 15  END IF;
 16  END T_personal;
 17  /

Trigger created.
```

---TRIGGER PARA INSERTAR O ELIMINAR TRANSPORTE NUEVO O DAÑADO

Create or replace trigger T_Transporte

After insert or delete on Transporte

For each row

BEGIN

IF INSERTING THEN

INSERT INTO registro_transporte (matricula,cap_carga, cod_trans, Status, Fecha)

VALUES (:new.matricula,:new.cap_carga, :new.cod_trans,'TRANSPORTE NUEVO', SYSDATE);

ELSIF DELETING THEN

INSERT INTO registro_transporte (matricula, cap_carga, cod_trans, Status, Fecha)

VALUES (:old.matricula,:old.cap_carga,:old.cod_trans, 'Transporte dañado', SYSDATE);

END IF;

END T_Transporte;

/

```
SQL> Create or replace trigger T_Transporte
  2
  3  After insert or delete  on Transporte
  4
  5  For each row
  6
  7  BEGIN
  8
  9  IF INSERTING THEN
 10
 11  INSERT INTO registro_transporte ( matricula,cap_carga, cod_trans
, Status, Fecha)
 12
 13    VALUES (:new.matricula,:new.cap_carga, :new.cod_trans,'TRANSPO
RTE NUEVO', SYSDATE);
 14
 15  ELSIF DELETING THEN
 16
 17  INSERT INTO registro_transporte ( matricula, cap_carga, cod_tran
s, Status, Fecha)
 18
 19    VALUES (:old.matricula,:old.cap_carga,:old.cod_trans, 'Transpo
rte dañado', SYSDATE);
 20
 21  END IF;
 22
 23  END T_Transporte;
 24
 25  /

Trigger created.
```

---TRIGGER PARA ACTUALIZAR LA TABLA TRANSPORTE

```
CREATE OR REPLACE TRIGGER   Act_Transporte
 AFTER INSERT ON Registro_transporte
for each row
 BEGIN
 INSERT INTO Transporte (matricula, cap_carga, cod_trans)
 VALUES (:new.matricula, :new.cap_carga, :new.cod_trans);
 END Act_transporte;
/
```

```
SQL> CREATE OR REPLACE TRIGGER   Act_Transporte
  2   AFTER UPDATE ON Registro_transporte
  3  FOR EACH ROW
  4   BEGIN
  5   INSERT INTO Transporte (matricula, cap_carga, cod_trans)
  6   VALUES (:new.matricula, :new.cap_carga, :new.cod_trans);
  7   END Act_transporte;
  8  /

Trigger created.
```

--TRIGGER PARA ACTUALIZAR CANTIDAD TOTAL DE INSUMOS

Create or replace trigger Act_Insumo

After Insert on Donador_insumo

For each row

BEGIN

UPDATE Insumo

SET cantidad_total =cantidad_total+:new.cantidad

WHERE cod_insumo = :new.cod_insumo;

END Act_Insumo;

/

```
SQL> Create or replace trigger Act_Insumo
  2  After  Insert  on Donador_insumo
  3  For each row
  4
  5  BEGIN
  6
  7  UPDATE Insumo
  8  SET cantidad_total =cantidad_total+:new.cantidad
  9  WHERE cod_insumo = :new.cod_insumo;
 10
 11  END Act_Insumo;
 12  /

Trigger created.
```

------VISTAS—

-------VISTA DE CENTRO DE DISTRIBUCIÓN-----

Create view vista_centro_distribucion as

select d.DNI_RUC,c.codigo_cedis

from donador d join donador_cedis x on d.DNI_RUC=x.DNI_RUC

Join centro_distribucion c on c.codigo_cedis = x.codigo_cedis;

```
SQL> Create view vista_centro_distribucion as
  2  select d.DNI_RUC,c.codigo_cedis
  3    from donador d join donador_cedis x on d.DNI_RUC=x.DNI_RUC
  4                   Join centro_distribucion c on c.codigo_cedi
s = x.codigo_cedis;

View created.
```

--------VISTA DE COD_DONACION---

Create view vista_cod_donacion as

select i.cod_donacion, d.DNI_RUC

from donador d join donador_insumo x on d.DNI_RUC=x.DNI_RUC

Join insumo i on i.cod_donacion = x.cod_donacion;

```
SQL> Create view vista_cod_donacion as
  2   select i.cod_donacion, d.DNI_RUC
  3     from donador d join donador_insumo x on d.DNI_RUC=x.DNI_RU
C
  4                   Join insumo i on i.cod_donacion = x.cod_don
acion;

View created.
```

-----VISTA DE TRANSPORTE

Create view vista_transporte as

select licencia, matricula

from repartidor;

```
SQL> Create view  vista_transporte as
  2  select licencia, matricula
  3     from repartidor;

View created.
```

----------VISTA DE PERSONAL

Create view vista_personal as

select i.cod_insumo, im.codigo_IM, p.DNI, x.fecha_dist, x.Cantidad_llevada

from INSUMO i join INSUMO_PERSONAL_IM x on i.cod_insumo=x.cod_insumo

Join Instalaciones_medicas im on im.codigo_IM = x.codigo_IM

join PERSONAL p on p.DNI=x.DNI;

```
SQL> Create view vista_personal as
  2   select i.cod_insumo, im.codigo_IM, p.DNI, x.fecha_dist, x.Ca
ntidad_llevada
  3     from INSUMO i join INSUMO_PERSONAL_IM x on i.cod_insumo=x.
cod_insumo
  4                   Join Instalaciones_medicas im on im.codigo_
IM = x.codigo_IM
  5    join PERSONAL p on p.DNI=x.DNI;

View created.
```

CONCLUSIÓN

El desarrollo de este proyecto nos ha permitido el desarrollo e implementación de los diferentes tipos de técnicas aprendidas durante el curso, de una manera más analítica ya que nos exigía de un poco más de compromiso y esfuerzo para el desarrollo del mismo, también se debe resaltar que esto nos ha abierto un panorama mucho mas amplio de cómo funcionan todos y cada uno de los aspectos aplicados en la implementación del esquema de base de datos ya que dependía casi al 100% de nosotros, y la importancia del estar preparados ante problemas adversos que puedan ocurrir, hemos logrado crear un nivel de conciencia de la importancia de que los centros médicos y el gobierno central, cuenten con una logística bien definida para el abastecimiento de los insumos que requiere, ya que hemos podido notar que sin un sistema de este tipo el sistema de salud nacional podría verse gravemente afectado.