UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN Y CONTROL

PROYECTO CÍSTICA

CURSO:

BASE DE DATOS II

PERTENECE A:

González, Eladio	8 – 972 – 2440
Delgado, José	9 – 759 – 550
Navarro, Diego	8 – 1002 – 2283
Mosquera, Alejandro	8 – 972 – 1615
Rodríguez, Víctor	20 – 70 – 7414

FACILITADOR:

ING. HENRY LEZCANO

PANAMÁ 2023

I. Proyecto Base de Datos I (Diseño).

- Nombre del proyecto de Base de Datos

  Proyecto Cística S.A.

- Misión de la Base de Datos:

  Diseñar e implementar una base de datos eficiente y robusta para Cística S.A., la empresa de envíos de productos farmacéuticos con drones. La base de datos debe ser capaz de almacenar y gestionar información detallada sobre drones, farmacias, envíos, centrales y personal. El diseño debe asegurar la integridad de los datos, permitir consultas eficientes y ofrecer soporte para el seguimiento de movimientos de drones, monitoreo de envíos y análisis estadístico.

- Objetivo de la misión de la Base de Datos:

  El objetivo principal de la base de datos es proporcionar a Cística S.A. una herramienta eficaz para gestionar sus operaciones de envío de productos farmacéuticos mediante drones. La base de datos deberá permitir:

  - Registro Detallado: Almacenar de manera precisa y completa la información sobre drones, farmacias, envíos, centrales y personal.

  - Movimiento de Drones: Registrar y rastrear los movimientos de los drones entre las diferentes centrales, incluyendo salidas, llegadas y cambios de central, con el fin de tener un control detallado sobre el movimiento de los activos.

  - Monitoreo de Envíos: Facilitar el monitoreo de los envíos, permitiendo asociar cada envío con el dron que lo realiza, la farmacia solicitante, la ubicación de entrega, peso, distancia, precio y tiempos de envío y entrega.

  - Análisis Estadístico: Proporcionar herramientas para realizar análisis estadísticos, permitiendo a Cística S.A. obtener información valiosa sobre el rendimiento de las

farmacias, eficiencia de los drones y otros aspectos operativos.

- Integridad de Datos: Garantizar la integridad de los datos almacenados mediante la implementación de restricciones y validaciones para prevenir inconsistencias o errores.

- Seguridad: Implementar medidas de seguridad para proteger la información confidencial y garantizar que solo personal autorizado tenga acceso a datos sensibles.

El éxito de la base de datos se medirá por su capacidad para proporcionar información precisa y oportuna, facilitar la toma de decisiones informadas y mejorar la eficiencia operativa de Cística S.A. en el ámbito de los envíos farmacéuticos con drones.

- Definición del Proyecto
  - Ámbito
    El ámbito del enunciado se centra en el diseño e implementación de una base de datos para la empresa Cística S.A., que se dedica al envío de productos farmacéuticos mediante el uso de drones. Incluye la gestión detallada de información relacionada con drones, farmacias, envíos, centrales y personal asociado con la operación. Además, abarca aspectos clave como el movimiento de drones, el monitoreo de envíos, el análisis estadístico y la seguridad de los datos.

  - Alcance o limites:
    1. Centrado en Operaciones de Envío Farmacéutico: El alcance se limita a las operaciones específicas de envío de productos farmacéuticos con drones. Otros aspectos del negocio de Cística S.A. que no estén directamente relacionados con estas operaciones pueden quedar fuera del alcance.

    2. Operaciones de Centrales y Drones: El diseño y seguimiento de las centrales y drones están incluidos, pero se limita a la información necesaria

para el monitoreo de movimientos y cambios de estado.

3. No Incluye Procesos de Fabricación de Drones: No se aborda la información relacionada con el proceso de fabricación de los drones, limitándose a los datos operativos.

4. Enfocado en Datos de Envíos Específicos: La base de datos se enfoca en el registro detallado de información relacionada con envíos específicos, como tiempos, distancias y precios. No incluye información detallada sobre los productos farmacéuticos en sí.

5. Análisis Estadístico Limitado: Aunque se menciona el análisis estadístico, el alcance no incluye análisis complejos o detallados que puedan requerir herramientas adicionales de análisis de datos.

6. Seguridad de Datos Básica: La implementación de medidas de seguridad se limita a proteger la información confidencial, pero no se profundiza en medidas avanzadas de seguridad informática.

7. No Incluye Aspectos Legales o Regulatorios: El alcance no abarca aspectos legales o regulaciones específicas relacionadas con el envío de productos farmacéuticos con drones.

- Análisis de requerimientos:
    1. Requisitos para la Información de Drones:
        - Código de dron (clave primaria).
        - Marca, modelo y año de fabricación.
        - Disponibilidad ('Y' o 'N').
        - Capacidad de carga y velocidad promedio.

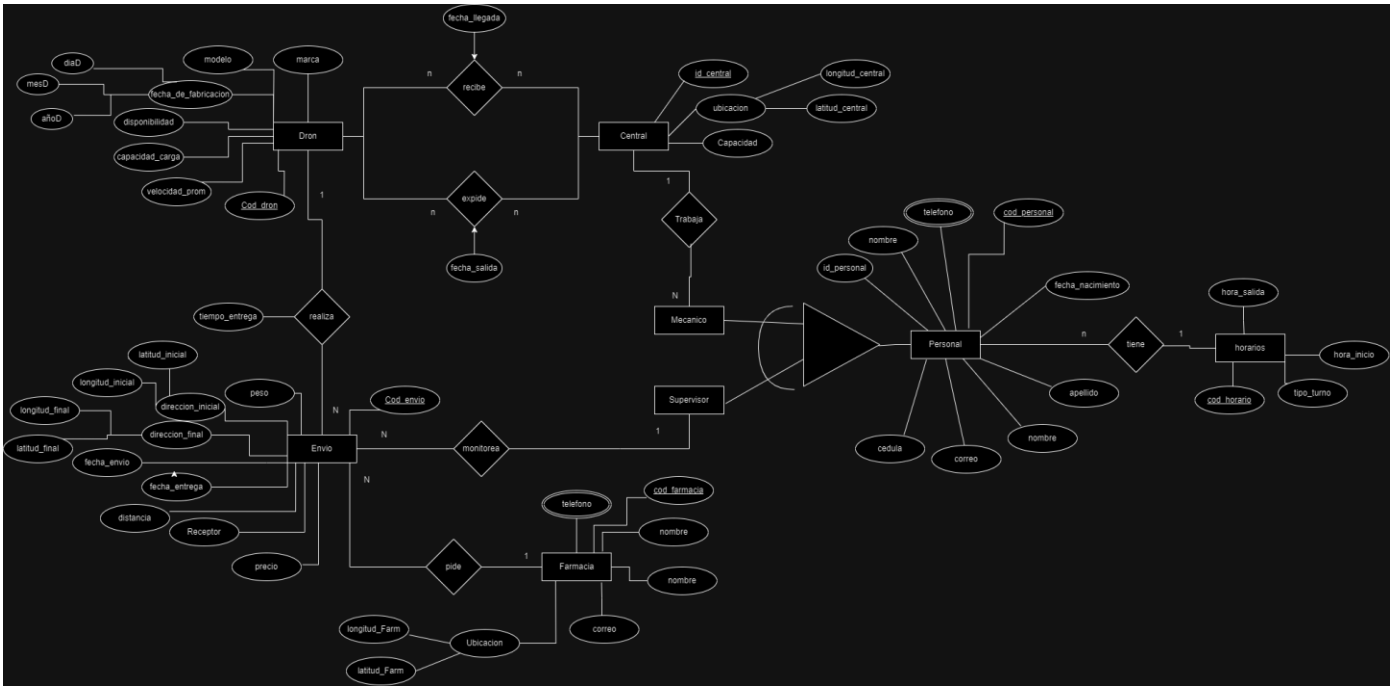
    2. Requisitos para la Información de Farmacias:
        - Código de farmacia (clave primaria).
        - RUC, nombre, correo.
        - Coordenadas decimales (latitud, longitud).
        - Teléfonos de contacto.

3. Requisitos para la Información de Envíos:
    - Código de envío (clave primaria).
    - Asociación con dron (clave foránea).
    - Asociación con farmacia (clave foránea).
    - Ubicación de entrega (latitud, longitud).
    - Peso, distancia, precio del envío.
    - Fechas y horas de salida y entrega.
    - Tiempo de entrega (derivado de fechas y horas).
4. Requisitos para la Información de Centrales:
    - Cod_central (clave primaria).
    - Nombre, longitud y latitud.
5. Requisitos para la Información de Personal:
    - Código de personal (clave primaria).
    - Nombre, apellido, cédula, fecha de nacimiento, sexo.
    - Teléfonos para cada tipo de personal.
    - Código de horario.
6. Requisitos para el Monitoreo de Drones:
    - Registro de movimientos entre centrales.
    - Registro de cambios de estado (disponibilidad).
7. Requisitos para el Análisis Estadístico:
    - Capacidades para calcular sumas y promedios.
    - Información estadística sobre precio, peso y distancia de los envíos por farmacia.
8. Requisitos de Seguridad:
    - Acceso restringido a personal autorizado.
    - Protección de información confidencial.
9. Requisitos de Integridad de Datos:
    - Restricciones para garantizar la coherencia y validez de los datos.

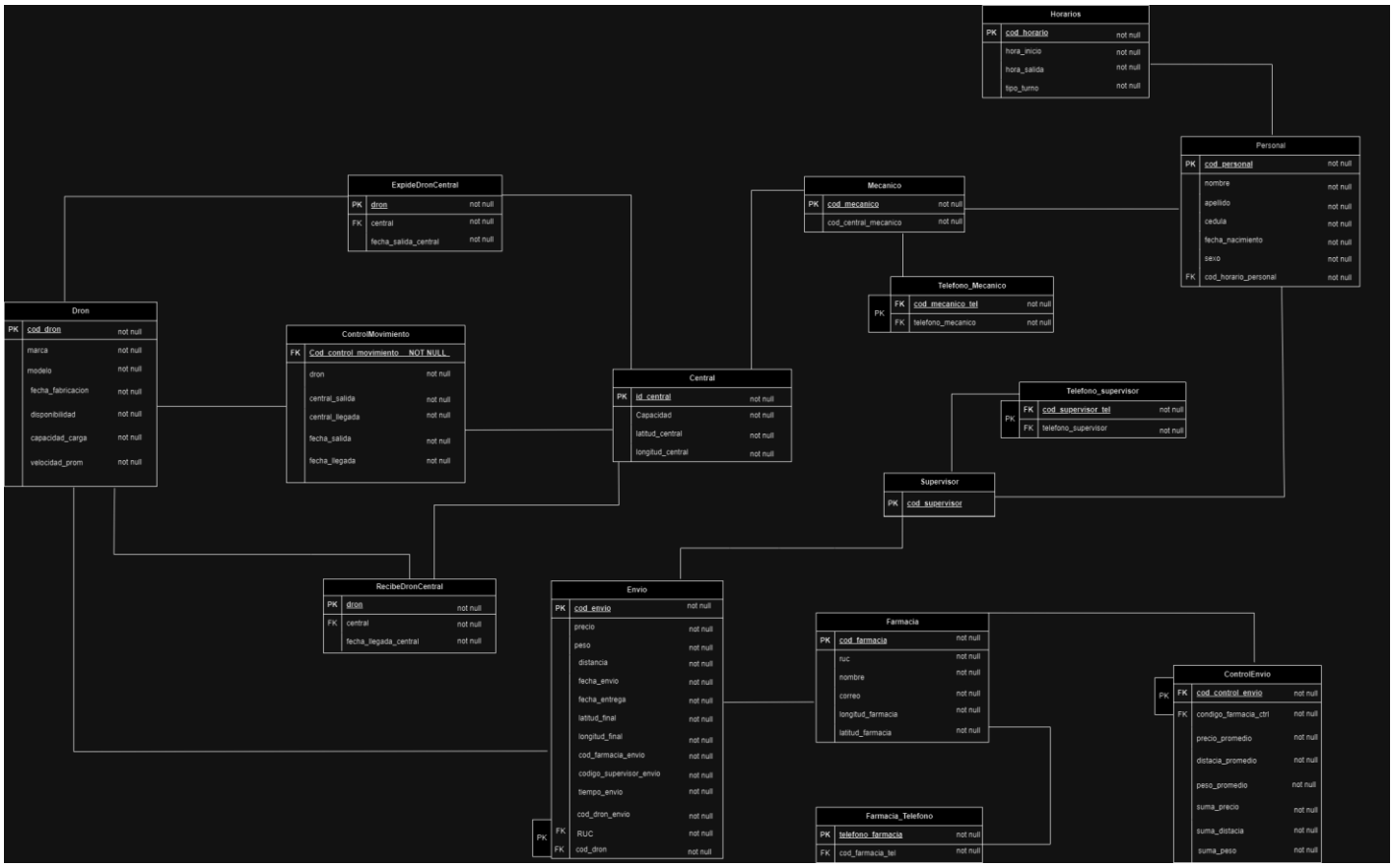
Requisitos adicionales:

- Soporte para los tres tipos de horarios: matutino, vespertino, nocturno.

- Modelado de la Base de Datos
  - Modelo Conceptual E/R



- Modelo Lógico Relación E/R Normalizado-- En Evaluación para IBDII



- Modelo Físico según sistema de gestión (Oracle)—En Evaluación para IBDII

```
CREATE TABLE Horarios (

    cod_horario VARCHAR2(4) NOT NULL,

    hora_inicio DATE NOT NULL, -- hora y minuto HH24, MIN

    hora_final DATE NOT NULL, -- hora y minuto HH24, MIN

    tipo_turno VARCHAR2(25) NOT NULL, -- matutino, vespertino, nocturno

    CONSTRAINT HORARIO_cod_horario_pk PRIMARY KEY (cod_horario)
```

```
);
CREATE SEQUENCE cod_central_seq START WITH 1 INCREMENT
BY 1;
CREATE TABLE Central(
    cod_central NUMBER NOT NULL,
    nombre VARCHAR2(25) UNIQUE,
    longitud_central VARCHAR2(50) NOT NULL,
    latitud_central VARCHAR2(50) NOT NULL,
    CONSTRAINT  CENTRAL_cond_central_pk  PRIMARY  KEY
(cod_central)
);
CREATE SEQUENCE cod_personal_seq START WITH 1 INCREMENT
BY 1;
CREATE TABLE Personal (
    cod_personal NUMBER,
    nombre VARCHAR2(50) NOT NULL,
    apellido VARCHAR2(50) NOT NULL,
    cedula VARCHAR2(10) UNIQUE NOT NULL,
    fecha_nacimiento DATE NOT NULL,
    sexo CHAR NOT NULL,
    cod_horario_personal VARCHAR2(4) NOT NULL,
    CONSTRAINT  PERSONAL_cod_horario_p_fk  FOREIGN  KEY
(cod_horario_personal)                          REFERENCES
Horarios(cod_horario),
    CONSTRAINT     PERSONAL_id_personal_pk     PRIMARY
KEY(cod_personal)
);
```

```
CREATE TABLE Mecanico (
    cod_mecanico NUMBER NOT NULL,
    cod_central_mecanico NUMBER NOT NULL,
    FOREIGN KEY (cod_mecanico) REFERENCES Personal(cod_personal),
    CONSTRAINT MECANICO_cod_central_FK FOREIGN KEY (cod_central_mecanico) REFERENCES Central(cod_central),
    CONSTRAINT MECANICO_cod_PERSONAL_PK PRIMARY KEY(cod_mecanico)
);
CREATE TABLE Supervisor (
    cod_supervisor NUMBER PRIMARY KEY,
    FOREIGN KEY (cod_supervisor) REFERENCES Personal(cod_personal)
);
CREATE TABLE Telefono_Mecanico(
    cod_mecanico_tel NUMBER NOT NULL,
    telefono_mecanico VARCHAR2(25) NOT NULL,
    CONSTRAINT TELEFONO_MECANICO_cod_mec_fk FOREIGN KEY (cod_mecanico_tel)
        REFERENCES Mecanico(cod_mecanico),
    CONSTRAINT Telefono_Mecanico_telefono_pk PRIMARY KEY (telefono_mecanico, cod_mecanico_tel)
);
CREATE TABLE Telefono_Supervisor(
    cod_supervisor_tel NUMBER NOT NULL,
    telefono_supervisor VARCHAR2(25) NOT NULL,
```

```
    CONSTRAINT    TEL_SUP_cod_sup_fk    FOREIGN    KEY
(cod_supervisor_tel)                        REFERENCES
Supervisor(cod_supervisor),

    CONSTRAINT   TEL_SUP_telefono_sup_PK   PRIMARY   KEY
(telefono_supervisor, cod_supervisor_tel)

);

CREATE SEQUENCE cod_farmacia_seq START WITH 1 INCREMENT
BY 1;

CREATE TABLE Farmacia(

    cod_farmacia NUMBER NOT NULL,

    ruc VARCHAR2(40) NOT NULL,

    nombre VARCHAR2(35) NOT NULL,

    correo VARCHAR2(55) NOT NULL UNIQUE,

    longitud_farmacia Number(8,6) NOT NULL,

    latitud_farmacia Number(8,6) NOT NULL,

    CONSTRAINT   FARMACIA_cod_farmacia   PRIMARY   KEY
(cod_farmacia)

);

CREATE TABLE Farmacia_Telefono(

    telefono_farmacia VARCHAR2(25),

    cod_farmacia_tel NUMBER NOT NULL,

    CONSTRAINT  FARMACIA_TEL_cod_farmacia  FOREIGN  KEY
(cod_farmacia_tel) REFERENCES Farmacia(cod_farmacia),

    CONSTRAINT      FARMACIA_PK      PRIMARY      KEY
(telefono_farmacia, cod_farmacia_tel)

);
```

```
CREATE SEQUENCE cod_dron_seq START WITH 1 INCREMENT BY
1;
CREATE TABLE Dron(
    cod_dron NUMBER NOT NULL,
    marca VARCHAR2(50) NOT NULL,
    modelo VARCHAR2(50) NOT NULL,
    fecha_fabricacion DATE NOT NULL,
    disponibilidad CHAR NOT NULL,
    capacidad_carga NUMBER NOT NULL, --en kilogramos
    velocidad_promedio NUMBER NOT NULL, -- en km/h ??
    CONSTRAINT DRON_cod_dron_pk PRIMARY KEY (cod_dron)
);
CREATE SEQUENCE cod_envio_seq START WITH 1 INCREMENT BY
1;
CREATE TABLE Envio(
    cod_envio NUMBER NOT NULL,
    precio NUMBER NOT NULL,
    distancia NUMBER(4,2) NOT NULL, -- kilometros
    fecha_envio TIMESTAMP(2), -- de cuando sale de la
farmacia
    fecha_entrega TIMESTAMP(2), --de cuando se entrega
al cliente
    peso NUMBER NOT NULL, --en kilogramos del paquete
que se envia
    latitud_final Number(8,6) NOT NULL, -- <-- para la
ubicacion final de la entrega
    longitud_final Number(8,6) NOT NULL,-- <-- para la
ubicacion final de la entrega
    cod_farmacia_envio NUMBER NOT NULL,
```

```
    cod_supervisor_envio NUMBER NOT NULL,
    cod_dron_envio NUMBER NOT NULL,
    tiempo_entrega INTERVAL DAY TO SECOND,        -- calcular
    CONSTRAINT ENVIO_cod_envio_pk PRIMARY KEY(cod_envio),
    CONSTRAINT ENVIO_cod_farm_envio_fk FOREIGN KEY (cod_farmacia_envio)
        REFERENCES Farmacia(cod_farmacia),
    CONSTRAINT ENVIO_cod_sup_envio_fk FOREIGN KEY (cod_supervisor_envio)
        REFERENCES Supervisor(cod_supervisor),
    CONSTRAINT ENVIO_cod_dron_envio_fk FOREIGN KEY(cod_dron_envio)
        REFERENCES Dron(cod_dron)
);
CREATE TABLE ExpideDronCentral (
    dron NUMBER NOT NULL,
    central NUMBER NOT NULL,
    fecha_salida_central TIMESTAMP(2) NOT NULL,
    CONSTRAINT EXPDRONCENT_dron_FK FOREIGN KEY (dron) REFERENCES Dron(cod_dron),
    CONSTRAINT EXPDRONCENT_central_FK FOREIGN KEY (central) REFERENCES Central(cod_central),
    CONSTRAINT EXPDRONCENT_PK PRIMARY KEY (dron, central, fecha_salida_central)
);
```

```
CREATE TABLE RecibeDronCentral (
    dron NUMBER NOT NULL,
    central NUMBER NOT NULL,
    fecha_llegada_central TIMESTAMP(2) NOT NULL,
    CONSTRAINT RECDRONCENT_dron_FK FOREIGN KEY (dron)
REFERENCES Dron(cod_dron),
    CONSTRAINT RECDRONCENT_central_FK FOREIGN KEY
(central) REFERENCES Central(cod_central),
    CONSTRAINT RECDRONCENT_PK PRIMARY KEY (dron,
central, fecha_llegada_central)
);
CREATE SEQUENCE cod_control_mov_seq START WITH 1
INCREMENT BY 1;
CREATE TABLE ControlMovimiento(
    cod_control_movimiento NUMBER NOT NULL,
    dron NUMBER NOT NULL,
    central_salida NUMBER NOT NULL,
    central_llegada NUMBER,
    fecha_salida TIMESTAMP(2) NOT NULL,
    fecha_llegada TIMESTAMP(2),
    CONSTRAINT CONTROL_central_llegada_FK FOREIGN
KEY(Central_llegada)
        REFERENCES Central(cod_central),
    CONSTRAINT CONTROL_central_salida_FK FOREIGN
KEY(Central_salida)
        REFERENCES Central(cod_central),
    CONSTRAINT CONTROL_cod_dron_ctrl_Fk FOREIGN
KEY(dron)
        REFERENCES Dron(cod_dron),
```

```
    CONSTRAINT      CONTROL_cod_control_PK      PRIMARY
KEY(cod_control_movimiento)

);

CREATE  SEQUENCE  cod_control_envio_seq  START  WITH  1
INCREMENT BY 1;

CREATE TABLE ControlEnvio (

    cod_control_envio NUMBER NOT NULL,

    cod_farmacia_ctrl NUMBER NOT NULL,

    precio_promedio NUMBER,

    distancia_promedio NUMBER,

    peso_promedio NUMBER,

    suma_precio NUMBER,

    suma_distancia NUMBER,

    suma_peso NUMBER,

    CONSTRAINT  CONTROLENVIO_farmacia_FK  FOREIGN  KEY
(cod_farmacia_ctrl)

        REFERENCES Farmacia(cod_farmacia),

    CONSTRAINT   CONTROLENVIO_cod_control_PK   PRIMARY
KEY(cod_control_envio, cod_farmacia_ctrl)

);
```

- Funciones

Con esta función se calcula el tiempo que dura la entrega desde que el dron sale de la farmacia hasta que entrega el dron entrega al cliente final

```sql
--funcion para calcular el tiempo recorrido despues de entregar el envio al cliente final
CREATE OR REPLACE FUNCTION calcularTiempoEntrega(
    p_fecha_envio IN Envio.fecha_envio%TYPE,
    p_fecha_entrega IN Envio.fecha_entrega%TYPE
) RETURN INTERVAL DAY TO SECOND IS
    v_tiempo_entrega INTERVAL DAY TO SECOND;
BEGIN
    v_tiempo_entrega := p_fecha_entrega - p_fecha_envio;
    RETURN v_tiempo_entrega;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error al calcular el tiempo de entrega.' || SQLERRM);
END calcularTiempoEntrega;
/
```

La siguiente función la empleamos para calcular la distancia geodésica desde la ubicación de la farmacia hasta la ubicación del cliente final

```sql
-- distancia aproximada
CREATE OR REPLACE FUNCTION RADIANS(degrees IN NUMBER)
 RETURN NUMBER IS
BEGIN
    RETURN degrees * 3.14159 / 180; -- Fórmula para convertir grados a radianes
END;
/
CREATE OR REPLACE FUNCTION calcularDistancia (
    P_latitud_inicial IN Farmacia.latitud_farmacia%TYPE,
    P_longitud_inicial IN Farmacia.longitud_farmacia%TYPE,
    P_latitud_final IN Envio.latitud_final%TYPE,
    P_longitud_final IN Envio.longitud_final%TYPE
) RETURN NUMBER IS
    earth_radius CONSTANT NUMBER := 6371; -- Radio de la Tierra en kilómetros
    d_lat NUMBER := RADIANS(P_latitud_final - P_latitud_inicial);
    d_lon NUMBER := RADIANS(P_longitud_final - P_longitud_inicial);
    a NUMBER;
    c NUMBER;
    distancia NUMBER;
BEGIN
    a := SIN(d_lat / 2) * SIN(d_lat / 2) +
         COS(RADIANS(P_latitud_inicial)) * COS(RADIANS(P_latitud_final)) *
         SIN(d_lon / 2) * SIN(d_lon / 2);
    c := 2 * ATAN2(SQRT(a), SQRT(1 - a));
    distancia := earth_radius * c;
    RETURN ROUND(distancia, 2);
EXCEPTION
    WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL CALCULAR LA DISTANCIA' || SQLERRM);

END;
/
```

También utilizamos una función para calcular los precios de los envíos en función de la distancia

```
53
54  CREATE or REPLACE FUNCTION calcularPrecio(
55    p_distancia IN Envio.distancia%TYPE
56  )
57  RETURN NUMBER IS
58  v_precio Envio.precio%TYPE;
59  BEGIN
60  IF p_distancia < 2 THEN
61    v_precio := 5;
62    ELSIF p_distancia BETWEEN 2 AND 5 THEN
63    v_precio := 10;
64    ELSIF p_distancia BETWEEN 5 AND 10 THEN
65    v_precio := 15;
66    ELSIF p_distancia BETWEEN 10 AND 20 THEN
67    v_precio := 20;
68    ELSE
69    v_precio := 35;
70  END IF;
71  RETURN v_precio;
72  EXCEPTION
73    WHEN OTHERS THEN
74        DBMS_OUTPUT.PUT_LINE('ERROR AL CALCULAR EL PRECIO' || SQLERRM);
75  END calcularPrecio;
76  /
77
```

- Procediimiento

Procedimiento para insertar los horarios

```
82
83  CREATE OR REPLACE PROCEDURE insertarHorario (
84    p_cod_horario IN Horarios.cod_horario%TYPE,
85    p_hora_inicio IN Horarios.hora_inicio%TYPE,
86    p_hora_final IN Horarios.hora_final%TYPE,
87    p_tipo_turno IN Horarios.tipo_turno%TYPE
88  ) AS
89  BEGIN
90  INSERT INTO Horarios (cod_horario, hora_inicio, hora_final, tipo_turno) VALUES
91  (p_cod_horario, p_hora_inicio, p_hora_final, p_tipo_turno);
92      DBMS_OUTPUT.PUT_LINE('Horario Insertado existosamente');
93
94  EXCEPTION
95    WHEN DUP_VAL_ON_INDEX THEN
96    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de horario');
97    WHEN OTHERS THEN
98        DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR HORARIOS');
99  COMMIT;
100 END insertarHorario;
101 /
```

Procedimiento para insertar las centrales

```
106
107 CREATE OR REPLACE PROCEDURE insertarCentral(
108     p_nombre IN Central.nombre%TYPE,
109     p_longitud_central IN Central.longitud_central%TYPE,
110     p_latitud_central IN Central.latitud_central%TYPE
111 ) AS
112 BEGIN
113     INSERT INTO Central(
114         cod_central,
115         nombre,
116         longitud_central,
117         latitud_central
118     ) VALUES (
119         cod_central_seq.NEXTVAL,
120         p_nombre,
121         p_longitud_central,
122         p_latitud_central
123     );
124     DBMS_OUTPUT.PUT_LINE('central Insertada existosamente');
125
126 EXCEPTION
127 WHEN DUP_VAL_ON_INDEX THEN
128   DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de central');
129   WHEN OTHERS THEN
130   DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CENTRAL');
131 COMMIT;
132 END insertarCentral;
133 /
134
```

Para insertar los mecánicos: se pasan los datos incluyendo el teléfono y la central donde trabaja

```sql
CREATE OR REPLACE PROCEDURE insertarMecanico (
    p_nombre IN Personal.nombre%TYPE,
    p_apellido IN Personal.apellido%TYPE,
    p_cedula IN Personal.cedula%TYPE,
    p_fecha_nacimiento IN Personal.fecha_nacimiento%TYPE,
    p_sexo IN Personal.sexo%TYPE,
    p_cod_horario_personal IN Personal.cod_horario_personal%TYPE,
    p_central_nombre IN Central.nombre%TYPE,
    p_telefono_mecanico IN Telefono_Mecanico.telefono_mecanico%TYPE
) IS
    v_cod_personal Personal.cod_personal%TYPE;
    v_cod_central Central.cod_central%TYPE;
BEGIN
SELECT cod_central INTO v_cod_central
  FROM Central WHERE nombre = p_central_nombre;


    INSERT INTO Personal (
        cod_personal,
        nombre,
        apellido,
        cedula,
        fecha_nacimiento,
        sexo,
        cod_horario_personal
    ) VALUES (
        cod_personal_seq.NEXTVAL,
        p_nombre,
        p_apellido,
        p_cedula,
        p_fecha_nacimiento,
        p_sexo,
        p_cod_horario_personal
    ) RETURNING cod_personal INTO v_cod_personal;


    INSERT INTO Mecanico (
        cod_mecanico,
        cod_central_mecanico
    ) VALUES (
        v_cod_personal,
        v_cod_central
    );

    INSERT INTO Telefono_Mecanico (
        cod_mecanico_tel,
        telefono_mecanico
    ) VALUES (
        v_cod_personal,
        p_telefono_mecanico
    );
    DBMS_OUTPUT.PUT_LINE('Mecanico Insertado existosamente');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('no se encontro un valor: ' || SQLERRM);
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de mecanico: ');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR MECANICO' || SQLERRM);
COMMIT;
END insertarMecanico;
/
```

Procedimiento para insertar a los supervisores que monitorean los envíos, los supervisores trabajan remotamente por eso no se indica central

```plsql
CREATE OR REPLACE PROCEDURE insertarSupervisor (
    p_nombre IN Personal.nombre%TYPE,
    p_apellido IN Personal.apellido%TYPE,
    p_cedula IN Personal.cedula%TYPE,
    p_fecha_nacimiento IN Personal.fecha_nacimiento%TYPE,
    p_sexo IN Personal.sexo%TYPE,
    p_cod_horario_personal IN Personal.cod_horario_personal%TYPE,
    p_telefono_supervisor IN Telefono_Supervisor.telefono_supervisor%TYPE
) IS
  v_cod_personal Personal.cod_personal%TYPE;
BEGIN

    INSERT INTO Personal (
        cod_personal,
        nombre,
        apellido,
        cedula,
        fecha_nacimiento,
        sexo,
        cod_horario_personal
    ) VALUES (
        cod_personal_seq.NEXTVAL,
        p_nombre,
        p_apellido,
        p_cedula,
        p_fecha_nacimiento,
        p_sexo,
        p_cod_horario_personal
    ) RETURNING cod_personal INTO v_cod_personal;

    INSERT INTO Supervisor (
        cod_supervisor
    ) VALUES (
        v_cod_personal
    );

    INSERT INTO Telefono_Supervisor(
      cod_supervisor_tel,
      telefono_supervisor
    ) VALUES (
      v_cod_personal,
      p_telefono_supervisor
    );
    DBMS_OUTPUT.PUT_LINE('Supervisor Insertado existosamente');

EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de supervisor');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR SUPREVISOR' || SQLERRM);
COMMIT;
END insertarSupervisor;
/
```

Procedimiento para insertar farmacias

```
68
69  CREATE OR REPLACE PROCEDURE insertarFarmacia(
70      p_ruc IN Farmacia.ruc%TYPE,
71      p_nombre IN Farmacia.nombre%TYPE,
72      p_correo IN Farmacia.correo%TYPE,
73      p_longitud_farmacia IN Farmacia.longitud_farmacia%TYPE,
74      p_latitud_farmacia IN Farmacia.latitud_farmacia%TYPE,
75      p_telefono_farmacia IN Farmacia_Telefono.telefono_farmacia%TYPE
76  ) IS
77      v_cod_farmacia Farmacia.cod_farmacia%TYPE;
78  BEGIN
79      INSERT INTO Farmacia(
80          cod_farmacia,
81          ruc,
82          nombre,
83          correo,
84          longitud_farmacia,
85          latitud_farmacia
86      ) VALUES (
87          cod_farmacia_seq.NEXTVAL,
88          p_ruc,
89          p_nombre,
90          p_correo,
91          p_longitud_farmacia,
92          p_latitud_farmacia
93      ) RETURNING cod_farmacia INTO v_cod_farmacia;
94
95      INSERT INTO Farmacia_Telefono(
96          telefono_farmacia,
97          cod_farmacia_tel
98      ) VALUES (
99          p_telefono_farmacia,
00          v_cod_farmacia
01      );
02          DBMS_OUTPUT.PUT_LINE('Farmacia Insertada existosamente');
03  EXCEPTION
04      WHEN DUP_VAL_ON_INDEX THEN
05          DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de farmacia');
06      WHEN OTHERS THEN
07          DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR FARMACIA');
08  COMMIT;
09  END insertarFarmacia;
10  /
```

Procedimiento para insertar los drones

```
CREATE OR REPLACE PROCEDURE insertarDron(
    p_marca IN Dron.marca%TYPE,
    p_modelo IN Dron.modelo%TYPE,
    p_fecha_fabricacion IN Dron.fecha_fabricacion%TYPE,
    p_disponibilidad IN Dron.disponibilidad%TYPE,
    p_capacidad_carga IN Dron.capacidad_carga%TYPE,
    p_velocidad_promedio IN Dron.velocidad_promedio%TYPE
) AS
BEGIN
    INSERT INTO Dron(
        cod_dron,
        marca,
        modelo,
        fecha_fabricacion,
        disponibilidad,
        capacidad_carga,
        velocidad_promedio
    ) VALUES (
        cod_dron_seq.NEXTVAL,
        p_marca,
        p_modelo,
        p_fecha_fabricacion,
        p_disponibilidad,
        p_capacidad_carga,
        p_velocidad_promedio
    );

    DBMS_OUTPUT.PUT_LINE('Dron Insertado existosamente');

EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de dron');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR DRON');
COMMIT;
END insertarDron;
/
```

Procedimiento para insertar los envios:

```
358
359 CREATE OR REPLACE PROCEDURE InsertarEnvio(
360     p_peso IN Envio.peso%TYPE,
361     p_latitud_final IN Envio.latitud_final %TYPE, --latitud final de la entrega
362     p_longitud_final IN Envio.longitud_final %TYPE,-- longitud final de la entrega
363     p_cod_farmacia_envio IN Farmacia.cod_farmacia%TYPE,
364     p_cod_supervisor_envio IN Supervisor.cod_supervisor%TYPE,
365     p_cod_dron_envio IN Dron.cod_dron%TYPE
366 ) IS
367     v_latitud_incial Farmacia.latitud_farmacia%TYPE;
368     v_longitud_incial Farmacia.longitud_farmacia%TYPE;
369     v_distancia Envio.distancia%TYPE;
370     v_precio Envio.precio%TYPE;
371 BEGIN
372     SELECT
373       latitud_farmacia,
374       longitud_farmacia
375     INTO
376       v_latitud_incial,
377       v_longitud_incial
378     FROM Farmacia
379     WHERE cod_farmacia = p_cod_farmacia_envio;
380
381     v_distancia := calcularDistancia(
382       v_latitud_incial,
383       v_longitud_incial,
384       p_latitud_final,
385       p_longitud_final
386     );
387
388     v_precio := calcularPrecio(
389       v_distancia
390     );
```

```
391
392     INSERT INTO Envio (
393         cod_envio,
394         precio,
395         distancia,
396         peso,
397         latitud_final,
398         longitud_final,
399         cod_farmacia_envio,
400         cod_supervisor_envio,
401         cod_dron_envio
402     ) VALUES (
403         cod_envio_seq.NEXTVAL,
404         v_precio,
405         v_distancia,
406         p_peso,
407         p_latitud_final,
408         p_longitud_final,
409         p_cod_farmacia_envio,
410         p_cod_supervisor_envio,
411         p_cod_dron_envio
412     );
413     DBMS_OUTPUT.PUT_LINE('envio Insertado existosamente');
414
415   EXCEPTION
416     WHEN DUP_VAL_ON_INDEX THEN
417       DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de envio');
418     WHEN OTHERS THEN
419       DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR ENVIO' || SQLERRM);
420   COMMIT;
421   END InsertarEnvio;
422   /
423
```

Procedimiento para insertar en la tabla expideDroncentral

```
CREATE OR REPLACE PROCEDURE expedirDron(
  p_dron Dron.cod_dron%TYPE,
  p_central Central.cod_central%TYPE
) IS
v_fecha_salida_central ExpideDronCentral.fecha_salida_central%TYPE;
BEGIN
v_fecha_salida_central := SYSDATE;
INSERT INTO ExpideDronCentral (
  dron,
  central,
  fecha_salida_central
) VALUES (
  p_dron,
  p_central,
  v_fecha_salida_central
);
    DBMS_OUTPUT.PUT_LINE('registro de salida de dron de la central '||p_central||' insertado correctamente');

EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de ExpidedronCentral');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR EN EXPIDE_DRON_CENTRAL');
COMMIT;
END expedirDron;
/
```

Procedimiento para cuando el dron llega a la farmacia

```
CREATE OR REPLACE PROCEDURE llegaFarmacia (
    p_cod_envio IN Envio.cod_envio%TYPE
) IS
BEGIN
    UPDATE Envio
    SET fecha_envio = SYSDATE
    WHERE cod_envio = p_cod_envio;

    DBMS_OUTPUT.PUT_LINE('Fecha de llegada a la farmacia actualizada exitosamente');

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No se encontró un envío con el código: ' || p_cod_envio);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error al actualizar la fecha de envío.');
COMMIT;
END llegaFarmacia;
/
```

Procedimiento para cuando llega el envio al cliente final

```
CREATE OR REPLACE PROCEDURE entregaPedido(
  p_cod_envio IN Envio.cod_envio%TYPE
) IS
BEGIN
  UPDATE Envio
  SET fecha_entrega = SYSDATE
  WHERE cod_envio = p_cod_envio;

  DBMS_OUTPUT.PUT_LINE('Fecha de entrega al cliente actualizada exitosamente');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No se encontro un envio con el codigo: ' || p_cod_envio);
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error al actualizar la fecha de envio.' );
COMMIT;
END entregaPedido;
/
```

Procedimiento para cuando el dron llega a una central

```
CREATE or REPLACE PROCEDURE recibirDron(
  p_dron Dron.cod_dron%TYPE,
  p_central Central.cod_central%TYPE
) IS
v_fecha_llegada_central RecibeDronCentral.fecha_llegada_central%TYPE;
BEGIN
v_fecha_llegada_central := SYSDATE;
INSERT INTO RecibeDronCentral (
  dron,
  central,
  fecha_llegada_central
) VALUES (
  p_dron,
  p_central,
  v_fecha_llegada_central
);
    DBMS_OUTPUT.PUT_LINE('registro de llegada de dron de la central '||p_central||' insertado correctamente');

EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de RecibeDronCentral');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR EN RECIBE_DRON_CENTRAL');
COMMIT;
END recibirDron;
/
```

Y procedimiento para actualizar los datos estadísticos de los envíos por farmacia

```
533
534  CREATE OR REPLACE PROCEDURE actualizarControlEnvios AS
535    CURSOR c_enviosPromedios IS
536      SELECT
537        cod_farmacia_envio as cod_farmacia,
538        AVG(precio) as precio_promedio,
539        AVG(distancia) as distancia_promedio,
540        AVG(peso) as peso_promedio,
541        SUM(precio) as suma_precio,
542        SUM(distancia) as suma_distancia,
543        SUM(peso) as suma_peso
544      FROM Envio
545      GROUP BY cod_farmacia_envio;
546  BEGIN
547    FOR r_enviosPromedio IN c_enviosPromedios LOOP
548      UPDATE ControlEnvio
549      SET
550        precio_promedio = r_enviosPromedio.precio_promedio,
551        distancia_promedio = r_enviosPromedio.distancia_promedio,
552        peso_promedio = r_enviosPromedio.peso_promedio,
553        suma_precio = r_enviosPromedio.suma_precio,
554        suma_distancia = r_enviosPromedio.suma_distancia,
555        suma_peso = r_enviosPromedio.suma_peso
556      WHERE cod_farmacia_ctrl = r_enviosPromedio.cod_farmacia;
557
558    END LOOP;
559        DBMS_OUTPUT.PUT_LINE('datos de envio por farmacia actualizados con exito');
560  EXCEPTION
561   WHEN NO_DATA_FOUND THEN
562      DBMS_OUTPUT.PUT_LINE('ERROR, NO SE ENCONTRÓ INFORMACIÓN');
563    WHEN OTHERS THEN
564      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR EN RECIBE_DRON_CENTRAL');
565  COMMIT;
566  END actualizarControlEnvios;
567  /
568
```

- Triggers

Triggers para insertal en la tabla control movimientos para saber cuando sale que dron de que central y cuando llega y a que central

```
CREATE OR REPLACE TRIGGER t_expide_dron_central
AFTER INSERT ON ExpideDronCentral
FOR EACH ROW
BEGIN
INSERT INTO ControlMovimiento (
  cod_control_movimiento,
  dron,
  central_salida,
  fecha_salida
  ) VALUES (
    cod_control_mov_seq.NEXTVAL,
    :new.dron,
    :new.central,
    :new.fecha_salida_central
  );
  EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('no se encontraron valores en el trigger t_expide_dron_central');
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de control_movimientos');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CONTROL_MOVIMIENTO');
  END t_expide_dron_central;
/

CREATE OR REPLACE TRIGGER t_recibe_dron_central
AFTER INSERT ON RecibeDronCentral
FOR EACH ROW
BEGIN
UPDATE ControlMovimiento
    SET
        central_llegada = :new.central,
        fecha_llegada = :new.fecha_llegada_central
    WHERE dron = :new.dron AND central_llegada IS NULL AND fecha_llegada IS NULL;
EXCEPTION
WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('no se encontraron valores en el trigger t_recibe_dron_central');
END t_recibe_dron_central;
/
```

Trigger para insertar una farmacia en la tabla de control_envio

```
CREATE OR REPLACE TRIGGER t_insertar_control_envio
AFTER INSERT ON Farmacia
FOR EACH ROW
BEGIN
INSERT INTO ControlEnvio (
  cod_control_envio,
  cod_farmacia_ctrl
) VALUES (
  cod_control_envio_seq.NEXTVAL,
  :new.cod_farmacia
);
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de control_envio');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CONTROL_ENVIO' || SQLERRM);
END t_insertar_control_envio;
/
```

Trigger para actualizar el tiempo de entrega cuando se le entrega el pedido al cliente

```
CREATE OR REPLACE TRIGGER t_tiempoEntrega
  BEFORE UPDATE OF fecha_entrega ON Envio
  FOR EACH ROW
  BEGIN
   :NEW.tiempo_entrega := calcularTiempoEntrega(:NEW.fecha_envio, :NEW.fecha_entrega);
EXCEPTION
 WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CONTROL_ENVIO' || SQLERRM);
END t_tiempoEntrega;
/
```

```
Connected.
SQL> CREATE TABLE Horarios (
  2      cod_horario VARCHAR2(4) NOT NULL,
  3      hora_inicio DATE NOT NULL, -- hora y minuto HH24, MIN
  4      hora_final DATE NOT NULL, -- hora y minuto HH24, MIN
  5      tipo_turno VARCHAR2(25) NOT NULL, -- matutino, vespertino, nocturno
  6      CONSTRAINT HORARIO_cod_horario_pk PRIMARY KEY (cod_horario)
  7  );

Table created.

SQL>
SQL>
SQL>
SQL> CREATE SEQUENCE cod_central_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE Central(
  2      cod_central NUMBER NOT NULL,
  3      nombre VARCHAR2(25) UNIQUE,
  4      longitud_central VARCHAR2(50) NOT NULL,
  5      latitud_central VARCHAR2(50) NOT NULL,
  6      CONSTRAINT CENTRAL_cond_central_pk PRIMARY KEY (cod_central)
  7  );

Table created.

SQL>
SQL>
SQL> CREATE SEQUENCE cod_personal_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE Personal (
  2      cod_personal NUMBER,
  3      nombre VARCHAR2(50) NOT NULL,
  4      apellido VARCHAR2(50) NOT NULL,
  5      cedula VARCHAR2(10) UNIQUE NOT NULL,
  6      fecha_nacimiento DATE NOT NULL,
  7      sexo CHAR NOT NULL,
  8      cod_horario_personal VARCHAR2(4) NOT NULL,
  9      CONSTRAINT PERSONAL_cod_horario_p_fk FOREIGN KEY (cod_horario_personal) REFERENCES Horarios(cod_horario),
 10      CONSTRAINT PERSONAL_id_personal_pk PRIMARY KEY(cod_personal)
 11  );

Table created.

SQL>
SQL> CREATE TABLE Mecanico (
  2      cod_mecanico NUMBER NOT NULL,
  3      cod_central_mecanico NUMBER NOT NULL,
  4      FOREIGN KEY (cod_mecanico) REFERENCES Personal(cod_personal),
  5      CONSTRAINT MECANICO_cod_central_FK FOREIGN KEY (cod_central_mecanico) REFERENCES Central(cod_central),
  6      CONSTRAINT MECANICO_cod_PERSONAL_PK PRIMARY KEY(cod_mecanico)
```

```
SQL>
SQL> CREATE TABLE Mecanico (
  2      cod_mecanico NUMBER NOT NULL,
  3      cod_central_mecanico NUMBER NOT NULL,
  4      FOREIGN KEY (cod_mecanico) REFERENCES Personal(cod_personal),
  5      CONSTRAINT MECANICO_cod_central_FK FOREIGN KEY (cod_central_mecanico) REFERENCES Central(cod_central),
  6      CONSTRAINT MECANICO_cod_PERSONAL_PK PRIMARY KEY(cod_mecanico)
  7  );

Table created.

SQL>
SQL> CREATE TABLE Supervisor (
  2      cod_supervisor NUMBER PRIMARY KEY,
  3      FOREIGN KEY (cod_supervisor) REFERENCES Personal(cod_personal)
  4  );

Table created.

SQL>
SQL>
SQL> CREATE TABLE Telefono_Mecanico(
  2      cod_mecanico_tel NUMBER NOT NULL,
  3      telefono_mecanico VARCHAR2(25) NOT NULL,
  4      CONSTRAINT TELEFONO_MECANICO_cod_mec_fk FOREIGN KEY (cod_mecanico_tel)
  5          REFERENCES Mecanico(cod_mecanico),
  6      CONSTRAINT Telefono_Mecanico_telefono_pk PRIMARY KEY (telefono_mecanico, cod_mecanico_tel)
  7  );

Table created.

SQL>
SQL>
SQL> CREATE TABLE Telefono_Supervisor(
  2      cod_supervisor_tel NUMBER NOT NULL,
  3      telefono_supervisor VARCHAR2(25) NOT NULL,
  4      CONSTRAINT TEL_SUP_cod_sup_fk FOREIGN KEY (cod_supervisor_tel) REFERENCES Supervisor(cod_supervisor),
  5      CONSTRAINT TEL_SUP_telefono_sup_PK PRIMARY KEY (telefono_supervisor, cod_supervisor_tel)
  6  );

Table created.

SQL>
SQL>
SQL> CREATE SEQUENCE cod_farmacia_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL>
SQL> CREATE TABLE Farmacia(
  2      cod_farmacia NUMBER NOT NULL,
  3      ruc VARCHAR2(40) NOT NULL,
  4      nombre VARCHAR2(35) NOT NULL,
  5      correo VARCHAR2(55) NOT NULL UNIQUE,
  6      longitud_farmacia Number(8,6) NOT NULL,
  7      latitud_farmacia Number(8,6) NOT NULL,
```

```
SQL>
SQL> CREATE TABLE Farmacia(
  2      cod_farmacia NUMBER NOT NULL,
  3      ruc VARCHAR2(40) NOT NULL,
  4      nombre VARCHAR2(35) NOT NULL,
  5      correo VARCHAR2(55) NOT NULL UNIQUE,
  6      longitud_farmacia Number(8,6) NOT NULL,
  7      latitud_farmacia Number(8,6) NOT NULL,
  8      CONSTRAINT FARMACIA_cod_farmacia PRIMARY KEY (cod_farmacia)
  9  );

Table created.

SQL>
SQL>
SQL> CREATE TABLE Farmacia_Telefono(
  2      telefono_farmacia VARCHAR2(25),
  3      cod_farmacia_tel NUMBER NOT NULL,
  4      CONSTRAINT FARMACIA_TEL_cod_farmacia FOREIGN KEY (cod_farmacia_tel) REFERENCES Farmacia(cod_farmacia),
  5      CONSTRAINT FARMACIA_PK PRIMARY KEY (telefono_farmacia, cod_farmacia_tel)
  6  );

Table created.

SQL>
SQL>
SQL> CREATE SEQUENCE cod_dron_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE Dron(
  2      cod_dron NUMBER NOT NULL,
  3      marca VARCHAR2(50) NOT NULL,
  4      modelo VARCHAR2(50) NOT NULL,
  5      fecha_fabricacion DATE NOT NULL,
  6      disponibilidad CHAR NOT NULL,
  7      capacidad_carga NUMBER NOT NULL, --en kilogramos
  8      velocidad_promedio NUMBER NOT NULL, -- en km/h ??
  9      CONSTRAINT DRON_cod_dron_pk PRIMARY KEY (cod_dron)
 10  );

Table created.

SQL>
SQL>
SQL> CREATE SEQUENCE cod_envio_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE Envio(
  2      cod_envio NUMBER NOT NULL,
  3      precio NUMBER NOT NULL,
  4      distancia NUMBER(4,2) NOT NULL, -- kilometros
```

```
SQL>
SQL>
SQL> CREATE SEQUENCE cod_envio_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE Envio(
  2      cod_envio NUMBER NOT NULL,
  3      precio NUMBER NOT NULL,
  4      distancia NUMBER(4,2) NOT NULL, -- kilometros
  5      fecha_envio TIMESTAMP(2), -- de cuando sale de la farmacia
  6      fecha_entrega TIMESTAMP(2), --de cuando se entrega al cliente
  7      peso NUMBER NOT NULL, --en kilogramos del paquete que se envia
  8      latitud_final Number(8,6) NOT NULL, -- <-- para la ubicacion final de la entrega
  9      longitud_final Number(8,6) NOT NULL,-- <-- para la ubicacion final de la entrega
 10      cod_farmacia_envio NUMBER NOT NULL,
 11      cod_supervisor_envio NUMBER NOT NULL,
 12      cod_dron_envio NUMBER NOT NULL,
 13      tiempo_entrega INTERVAL DAY TO SECOND,     --calcular
 14      CONSTRAINT ENVIO_cod_envio_pk PRIMARY KEY(cod_envio),
 15      CONSTRAINT ENVIO_cod_farm_envio_fk FOREIGN KEY (cod_farmacia_envio)
 16          REFERENCES Farmacia(cod_farmacia),
 17      CONSTRAINT ENVIO_cod_sup_envio_fk FOREIGN KEY (cod_supervisor_envio)
 18          REFERENCES Supervisor(cod_supervisor),
 19      CONSTRAINT ENVIO_cod_dron_envio_fk FOREIGN KEY(cod_dron_envio)
 20          REFERENCES Dron(cod_dron)
 21  );

Table created.

SQL>
SQL> CREATE TABLE ExpideDronCentral (
  2      dron NUMBER NOT NULL,
  3      central NUMBER NOT NULL,
  4      fecha_salida_central TIMESTAMP(2) NOT NULL,
  5      CONSTRAINT EXPDRONCENT_dron_FK FOREIGN KEY (dron) REFERENCES Dron(cod_dron),
  6      CONSTRAINT EXPDRONCENT_central_FK FOREIGN KEY (central) REFERENCES Central(cod_central),
  7      CONSTRAINT EXPDRONCENT_PK PRIMARY KEY (dron, central, fecha_salida_central)
  8  );

Table created.

SQL>
SQL> CREATE TABLE RecibeDronCentral (
  2      dron NUMBER NOT NULL,
  3      central NUMBER NOT NULL,
  4      fecha_llegada_central TIMESTAMP(2) NOT NULL,
  5      CONSTRAINT RECDRONCENT_dron_FK FOREIGN KEY (dron) REFERENCES Dron(cod_dron),
  6      CONSTRAINT RECDRONCENT_central_FK FOREIGN KEY (central) REFERENCES Central(cod_central),
  7      CONSTRAINT RECDRONCENT_PK PRIMARY KEY (dron, central, fecha_llegada_central)
  8  );

Table created.
```

```
SQL>
SQL> CREATE SEQUENCE cod_control_mov_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE ControlMovimiento(
  2      cod_control_movimiento NUMBER NOT NULL,
  3      dron NUMBER NOT NULL,
  4      central_salida NUMBER NOT NULL,
  5      central_llegada NUMBER,
  6      fecha_salida TIMESTAMP(2) NOT NULL,
  7      fecha_llegada TIMESTAMP(2),
  8      CONSTRAINT CONTROL_central_llegada_FK FOREIGN KEY(Central_llegada)
  9          REFERENCES Central(cod_central),
 10      CONSTRAINT CONTROL_central_salida_FK FOREIGN KEY(Central_salida)
 11          REFERENCES Central(cod_central),
 12      CONSTRAINT CONTROL_cod_dron_ctrl_Fk FOREIGN KEY(dron)
 13          REFERENCES Dron(cod_dron),
 14      CONSTRAINT CONTROL_cod_control_PK PRIMARY KEY(cod_control_movimiento)
 15  );

Table created.

SQL>
SQL> CREATE SEQUENCE cod_control_envio_seq START WITH 1 INCREMENT BY 1;

Sequence created.

SQL> CREATE TABLE ControlEnvio (
  2      cod_control_envio NUMBER NOT NULL,
  3      cod_farmacia_ctrl NUMBER NOT NULL,
  4      precio_promedio NUMBER,
  5      distancia_promedio NUMBER,
  6      peso_promedio NUMBER,
  7      suma_precio NUMBER,
  8      suma_distancia NUMBER,
  9      suma_peso NUMBER,
 10      CONSTRAINT CONTROLENVIO_farmacia_FK FOREIGN KEY (cod_farmacia_ctrl)
 11          REFERENCES Farmacia(cod_farmacia),
 12      CONSTRAINT CONTROLENVIO_cod_control_PK PRIMARY KEY(cod_control_envio, cod_farmacia_ctrl)
 13  );

Table created.
```

```
SQL> ------------------------------------------------------------------------------------------------------------------------
SQL>
SQL> --funcion para calcular el tiempo recorrido despues de entregar el envio al cliente final
SQL> CREATE OR REPLACE FUNCTION calcularTiempoEntrega(
  2      p_fecha_envio IN Envio.fecha_envio%TYPE,
  3      p_fecha_entrega IN Envio.fecha_entrega%TYPE
  4  ) RETURN INTERVAL DAY TO SECOND IS
  5      v_tiempo_entrega INTERVAL DAY TO SECOND;
  6  BEGIN
  7      v_tiempo_entrega := p_fecha_entrega - p_fecha_envio;
  8      RETURN v_tiempo_entrega;
  9  EXCEPTION
 10      WHEN OTHERS THEN
 11          DBMS_OUTPUT.PUT_LINE('Error al calcular el tiempo de entrega.' || SQLERRM);
 12  END calcularTiempoEntrega;
 13  /

Function created.

SQL>
SQL> -- distancia aproximada
SQL> CREATE OR REPLACE FUNCTION RADIANS(degrees IN NUMBER)
  2   RETURN NUMBER IS
  3   BEGIN
  4       RETURN degrees * 3.14159 / 180; -- Fórmula para convertir grados a radianes
  5   END;
  6   /

Function created.

SQL> CREATE OR REPLACE FUNCTION calcularDistancia (
  2      P_latitud_inicial IN Farmacia.latitud_farmacia%TYPE,
  3      P_longitud_inicial IN Farmacia.longitud_farmacia%TYPE,
  4      P_latitud_final IN Envio.latitud_final%TYPE,
  5      P_longitud_final IN Envio.longitud_final%TYPE
  6  ) RETURN NUMBER IS
  7      earth_radius CONSTANT NUMBER := 6371; -- Radio de la Tierra en kilómetros
  8      d_lat NUMBER := RADIANS(P_latitud_final - P_latitud_inicial);
  9      d_lon NUMBER := RADIANS(P_longitud_final - P_longitud_inicial);
 10      a NUMBER;
 11      c NUMBER;
 12      distancia NUMBER;
 13  BEGIN
 14      a := SIN(d_lat / 2) * SIN(d_lat / 2) +
 15           COS(RADIANS(P_latitud_inicial)) * COS(RADIANS(P_latitud_final)) *
 16           SIN(d_lon / 2) * SIN(d_lon / 2);
 17      c := 2 * ATAN2(SQRT(a), SQRT(1 - a));
 18      distancia := earth_radius * c;
 19      RETURN ROUND(distancia, 2);
 20  EXCEPTION
 21      WHEN OTHERS THEN
 22      DBMS_OUTPUT.PUT_LINE('ERROR AL CALCULAR LA DISTANCIA' || SQLERRM);
 23
 24  END;
 25  /

Function created.
```

```
SQL>
SQL> CREATE or REPLACE FUNCTION calcularPrecio(
  2    p_distancia IN Envio.distancia%TYPE
  3  )
  4  RETURN NUMBER IS
  5  v_precio Envio.precio%TYPE;
  6  BEGIN
  7  IF p_distancia < 2 THEN
  8    v_precio := 5;
  9    ELSIF p_distancia BETWEEN 2 AND 5 THEN
 10    v_precio := 10;
 11    ELSIF p_distancia BETWEEN 5 AND 10 THEN
 12    v_precio := 15;
 13    ELSIF p_distancia BETWEEN 10 AND 20 THEN
 14    v_precio := 20;
 15    ELSE
 16    v_precio := 35;
 17  END IF;
 18  RETURN v_precio;
 19  EXCEPTION
 20    WHEN OTHERS THEN
 21        DBMS_OUTPUT.PUT_LINE('ERROR AL CALCULAR EL PRECIO' || SQLERRM);
 22  END calcularPrecio;
 23  /

Function created.

SQL>
SQL>
SQL> ----------------------------------------------------------------------------------------------------------------
SQL> ------------------------------------------------               Insertar Horario
SQL> ----------------------------------------------------------------------------------------------------------------
SQL>
SQL> CREATE OR REPLACE PROCEDURE insertarHorario (
  2    p_cod_horario IN Horarios.cod_horario%TYPE,
  3    p_hora_inicio IN Horarios.hora_inicio%TYPE,
  4    p_hora_final IN Horarios.hora_final%TYPE,
  5    p_tipo_turno IN Horarios.tipo_turno%TYPE
  6  ) AS
  7  BEGIN
  8  INSERT INTO Horarios (cod_horario, hora_inicio, hora_final, tipo_turno) VALUES
  9  (p_cod_horario, p_hora_inicio, p_hora_final, p_tipo_turno);
 10      DBMS_OUTPUT.PUT_LINE('Horario Insertado existosamente');
 11
 12  EXCEPTION
 13    WHEN DUP_VAL_ON_INDEX THEN
 14    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de horario');
 15    WHEN OTHERS THEN
 16        DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR HORARIOS');
 17  COMMIT;
 18  END insertarHorario;
 19  /

Procedure created.

SQL>
```

```
SQL>
SQL> CREATE OR REPLACE PROCEDURE insertarCentral(
  2      p_nombre IN Central.nombre%TYPE,
  3      p_longitud_central IN Central.longitud_central%TYPE,
  4      p_latitud_central IN Central.latitud_central%TYPE
  5  ) AS
  6  BEGIN
  7      INSERT INTO Central(
  8          cod_central,
  9          nombre,
 10          longitud_central,
 11          latitud_central
 12      ) VALUES (
 13          cod_central_seq.NEXTVAL,
 14          p_nombre,
 15          p_longitud_central,
 16          p_latitud_central
 17      );
 18      DBMS_OUTPUT.PUT_LINE('central Insertada existosamente');
 19
 20  EXCEPTION
 21  WHEN DUP_VAL_ON_INDEX THEN
 22    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de central');
 23    WHEN OTHERS THEN
 24    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CENTRAL');
 25  COMMIT;
 26  END insertarCentral;
 27  /

Procedure created.
```

```
    v_cod_personal Personal.cod_personal%TYPE;
    v_cod_central Central.cod_central%TYPE;
BEGIN
SELECT cod_central INTO v_cod_central
   FROM Central WHERE nombre = p_central_nombre;


    INSERT INTO Personal (
        cod_personal,
        nombre,
        apellido,
        cedula,
        fecha_nacimiento,
        sexo,
        cod_horario_personal
    ) VALUES (
        cod_personal_seq.NEXTVAL,
        p_nombre,
        p_apellido,
        p_cedula,
        p_fecha_nacimiento,
        p_sexo,
        p_cod_horario_personal
    ) RETURNING cod_personal INTO v_cod_personal;


    INSERT INTO Mecanico (
        cod_mecanico,
        cod_central_mecanico
    ) VALUES (
        v_cod_personal,
        v_cod_central
    );

    INSERT INTO Telefono_Mecanico (
        cod_mecanico_tel,
        telefono_mecanico
    ) VALUES (
        v_cod_personal,
        p_telefono_mecanico
    );
    DBMS_OUTPUT.PUT_LINE('Mecanico Insertado existosamente');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('no se encontro un valor: ' || SQLERRM);
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de mecanico: ');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR MECANICO' || SQLERRM);
COMMIT;
END insertarMecanico;
/

Procedure created.
```

```
SQL> CREATE OR REPLACE PROCEDURE insertarSupervisor (
  2      p_nombre IN Personal.nombre%TYPE,
  3      p_apellido IN Personal.apellido%TYPE,
  4      p_cedula IN Personal.cedula%TYPE,
  5      p_fecha_nacimiento IN Personal.fecha_nacimiento%TYPE,
  6      p_sexo IN Personal.sexo%TYPE,
  7      p_cod_horario_personal IN Personal.cod_horario_personal%TYPE,
  8      p_telefono_supervisor IN Telefono_Supervisor.telefono_supervisor%TYPE
  9  ) IS
 10    v_cod_personal Personal.cod_personal%TYPE;
 11  BEGIN
 12
 13      INSERT INTO Personal (
 14          cod_personal,
 15          nombre,
 16          apellido,
 17          cedula,
 18          fecha_nacimiento,
 19          sexo,
 20          cod_horario_personal
 21      ) VALUES (
 22          cod_personal_seq.NEXTVAL,
 23          p_nombre,
 24          p_apellido,
 25          p_cedula,
 26          p_fecha_nacimiento,
 27          p_sexo,
 28          p_cod_horario_personal
 29      ) RETURNING cod_personal INTO v_cod_personal;
 30
 31      INSERT INTO Supervisor (
 32          cod_supervisor
 33      ) VALUES (
 34          v_cod_personal
 35      );
 36
 37      INSERT INTO Telefono_Supervisor(
 38        cod_supervisor_tel,
 39        telefono_supervisor
 40      ) VALUES (
 41        v_cod_personal,
 42        p_telefono_supervisor
 43      );
 44      DBMS_OUTPUT.PUT_LINE('Supervisor Insertado existosamente');
 45
 46  EXCEPTION
 47    WHEN DUP_VAL_ON_INDEX THEN
 48      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de supervisor');
 49    WHEN OTHERS THEN
 50      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR SUPREVISOR' || SQLERRM);
 51  COMMIT;
 52  END insertarSupervisor;
 53  /

Procedure created.
```

```
SQL>
SQL> CREATE OR REPLACE PROCEDURE insertarFarmacia(
  2    p_ruc IN Farmacia.ruc%TYPE,
  3    p_nombre IN Farmacia.nombre%TYPE,
  4    p_correo IN Farmacia.correo%TYPE,
  5    p_longitud_farmacia IN Farmacia.longitud_farmacia%TYPE,
  6    p_latitud_farmacia IN Farmacia.latitud_farmacia%TYPE,
  7    p_telefono_farmacia IN Farmacia_Telefono.telefono_farmacia%TYPE
  8  ) IS
  9    v_cod_farmacia Farmacia.cod_farmacia%TYPE;
 10  BEGIN
 11    INSERT INTO Farmacia(
 12      cod_farmacia,
 13      ruc,
 14      nombre,
 15      correo,
 16      longitud_farmacia,
 17      latitud_farmacia
 18    ) VALUES (
 19      cod_farmacia_seq.NEXTVAL,
 20      p_ruc,
 21      p_nombre,
 22      p_correo,
 23      p_longitud_farmacia,
 24      p_latitud_farmacia
 25    ) RETURNING cod_farmacia INTO v_cod_farmacia;
 26
 27    INSERT INTO Farmacia_Telefono(
 28      telefono_farmacia,
 29      cod_farmacia_tel
 30    ) VALUES (
 31      p_telefono_farmacia,
 32      v_cod_farmacia
 33    );
 34      DBMS_OUTPUT.PUT_LINE('Farmacia Insertada existosamente');
 35  EXCEPTION
 36    WHEN DUP_VAL_ON_INDEX THEN
 37      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de farmacia');
 38    WHEN OTHERS THEN
 39      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR FARMACIA');
 40  COMMIT;
 41  END insertarFarmacia;
 42  /

Procedure created.
```

```
SQL>
SQL>
SQL> CREATE OR REPLACE PROCEDURE insertarDron(
  2      p_marca IN Dron.marca%TYPE,
  3      p_modelo IN Dron.modelo%TYPE,
  4      p_fecha_fabricacion IN Dron.fecha_fabricacion%TYPE,
  5      p_disponibilidad IN Dron.disponibilidad%TYPE,
  6      p_capacidad_carga IN Dron.capacidad_carga%TYPE,
  7      p_velocidad_promedio IN Dron.velocidad_promedio%TYPE
  8  ) AS
  9  BEGIN
 10      INSERT INTO Dron(
 11          cod_dron,
 12          marca,
 13          modelo,
 14          fecha_fabricacion,
 15          disponibilidad,
 16          capacidad_carga,
 17          velocidad_promedio
 18      ) VALUES (
 19          cod_dron_seq.NEXTVAL,
 20          p_marca,
 21          p_modelo,
 22          p_fecha_fabricacion,
 23          p_disponibilidad,
 24          p_capacidad_carga,
 25          p_velocidad_promedio
 26      );
 27
 28      DBMS_OUTPUT.PUT_LINE('Dron Insertado existosamente');
 29
 30  EXCEPTION
 31    WHEN DUP_VAL_ON_INDEX THEN
 32      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de dron');
 33    WHEN OTHERS THEN
 34      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR DRON');
 35  COMMIT;
 36  END insertarDron;
 37  /

Procedure created.
```

```
SQL>
SQL> CREATE OR REPLACE PROCEDURE InsertarEnvio(
  2      p_peso IN Envio.peso%TYPE,
  3      p_latitud_final IN Envio.latitud_final %TYPE, --latitud final de la entrega
  4      p_longitud_final IN Envio.longitud_final %TYPE,-- longitud final de la entrega
  5      p_cod_farmacia_envio IN Farmacia.cod_farmacia%TYPE,
  6      p_cod_supervisor_envio IN Supervisor.cod_supervisor%TYPE,
  7      p_cod_dron_envio IN Dron.cod_dron%TYPE
  8  ) IS
  9      v_latitud_incial Farmacia.latitud_farmacia%TYPE;
 10      v_longitud_incial Farmacia.longitud_farmacia%TYPE;
 11      v_distancia Envio.distancia%TYPE;
 12      v_precio Envio.precio%TYPE;
 13  BEGIN
 14      SELECT
 15        latitud_farmacia,
 16        longitud_farmacia
 17      INTO
 18        v_latitud_incial,
 19        v_longitud_incial
 20      FROM Farmacia
 21      WHERE cod_farmacia = p_cod_farmacia_envio;
 22
 23      v_distancia := calcularDistancia(
 24        v_latitud_incial,
 25        v_longitud_incial,
 26        p_latitud_final,
 27        p_longitud_final
 28      );
 29
 30      v_precio := calcularPrecio(
 31        v_distancia
 32      );
 33
```

```
33
34      INSERT INTO Envio (
35          cod_envio,
36          precio,
37          distancia,
38          peso,
39          latitud_final,
40          longitud_final,
41          cod_farmacia_envio,
42          cod_supervisor_envio,
43          cod_dron_envio
44      ) VALUES (
45          cod_envio_seq.NEXTVAL,
46          v_precio,
47          v_distancia,
48          p_peso,
49          p_latitud_final,
50          p_longitud_final,
51          p_cod_farmacia_envio,
52          p_cod_supervisor_envio,
53          p_cod_dron_envio
54      );
55      DBMS_OUTPUT.PUT_LINE('envio Insertado existosamente');
56
57  EXCEPTION
58    WHEN DUP_VAL_ON_INDEX THEN
59      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de envio');
60    WHEN OTHERS THEN
61      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR ENVIO' || SQLERRM);
62  COMMIT;
63  END InsertarEnvio;
64  /

Procedure created.
```

```
SQL> ------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE llegaFarmacia (
  2      p_cod_envio IN Envio.cod_envio%TYPE
  3  ) IS
  4  BEGIN
  5      UPDATE Envio
  6      SET fecha_envio = SYSDATE
  7      WHERE cod_envio = p_cod_envio;
  8
  9      DBMS_OUTPUT.PUT_LINE('Fecha de llegada a la farmacia actualizada exitosamente');
 10
 11  EXCEPTION
 12      WHEN NO_DATA_FOUND THEN
 13          DBMS_OUTPUT.PUT_LINE('No se encontró un envío con el código: ' || p_cod_envio);
 14      WHEN OTHERS THEN
 15          DBMS_OUTPUT.PUT_LINE('Error al actualizar la fecha de envío.');
 16  COMMIT;
 17  END llegaFarmacia;
 18  /

 Procedure created.
```

```
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE entregaPedido(
  2    p_cod_envio IN Envio.cod_envio%TYPE
  3  ) IS
  4  BEGIN
  5    UPDATE Envio
  6    SET fecha_entrega = SYSDATE
  7    WHERE cod_envio = p_cod_envio;
  8
  9    DBMS_OUTPUT.PUT_LINE('Fecha de entrega al cliente actualizada exitosamente');
 10
 11  EXCEPTION
 12    WHEN NO_DATA_FOUND THEN
 13      DBMS_OUTPUT.PUT_LINE('No se encontro un envio con el codigo: ' || p_cod_envio);
 14    WHEN OTHERS THEN
 15      DBMS_OUTPUT.PUT_LINE('Error al actualizar la fecha de envio.' );
 16  COMMIT;
 17  END entregaPedido;
 18  /

Procedure created.

SQL>
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------
SQL> ----------------                    Insertar cuando el dron sale de una central                                    ------------
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------
SQL>
SQL> CREATE OR REPLACE PROCEDURE expedirDron(
  2    p_dron Dron.cod_dron%TYPE,
  3    p_central Central.cod_central%TYPE
  4  ) IS
  5  v_fecha_salida_central ExpideDronCentral.fecha_salida_central%TYPE;
  6  BEGIN
  7  v_fecha_salida_central := SYSDATE;
  8  INSERT INTO ExpideDronCentral (
  9    dron,
 10    central,
 11    fecha_salida_central
 12  ) VALUES (
 13    p_dron,
 14    p_central,
 15    v_fecha_salida_central
 16  );
 17      DBMS_OUTPUT.PUT_LINE('registro de salida de dron de la central '||p_central||' insertado correctamente');
 18
 19  EXCEPTION
 20    WHEN DUP_VAL_ON_INDEX THEN
 21      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de ExpidedronCentral');
 22    WHEN OTHERS THEN
 23      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR EN EXPIDE_DRON_CENTRAL');
 24  COMMIT;
 25  END expedirDron;
 26  /

Procedure created.
```

```
SQL>
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------
SQL> ----------------                    Insertar cuando el dron llega a una central
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------
SQL> CREATE or REPLACE PROCEDURE recibirDron(
  2    p_dron Dron.cod_dron%TYPE,
  3    p_central Central.cod_central%TYPE
  4  ) IS
  5  v_fecha_llegada_central RecibeDronCentral.fecha_llegada_central%TYPE;
  6  BEGIN
  7  v_fecha_llegada_central := SYSDATE;
  8  INSERT INTO RecibeDronCentral (
  9    dron,
 10    central,
 11    fecha_llegada_central
 12  ) VALUES (
 13    p_dron,
 14    p_central,
 15    v_fecha_llegada_central
 16  );
 17      DBMS_OUTPUT.PUT_LINE('registro de llegada de dron de la central '||p_central||' insertado correctamente');
 18
 19  EXCEPTION
 20    WHEN DUP_VAL_ON_INDEX THEN
 21      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de RecibeDronCentral');
 22    WHEN OTHERS THEN
 23      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR EN RECIBE_DRON_CENTRAL');
 24  COMMIT;
 25  END recibirDron;
 26  /

Procedure created.
```

```
SQL>
SQL> CREATE OR REPLACE PROCEDURE actualizarControlEnvios AS
  2    CURSOR c_enviosPromedios IS
  3      SELECT
  4        cod_farmacia_envio as cod_farmacia,
  5        AVG(precio) as precio_promedio,
  6        AVG(distancia) as distancia_promedio,
  7        AVG(peso) as peso_promedio,
  8        SUM(precio) as suma_precio,
  9        SUM(distancia) as suma_distancia,
 10        SUM(peso) as suma_peso
 11      FROM Envio
 12      GROUP BY cod_farmacia_envio;
 13  BEGIN
 14    FOR r_enviosPromedio IN c_enviosPromedios LOOP
 15      UPDATE ControlEnvio
 16      SET
 17        precio_promedio = r_enviosPromedio.precio_promedio,
 18        distancia_promedio = r_enviosPromedio.distancia_promedio,
 19        peso_promedio = r_enviosPromedio.peso_promedio,
 20        suma_precio = r_enviosPromedio.suma_precio,
 21        suma_distancia = r_enviosPromedio.suma_distancia,
 22        suma_peso = r_enviosPromedio.suma_peso
 23      WHERE cod_farmacia_ctrl = r_enviosPromedio.cod_farmacia;
 24
 25    END LOOP;
 26        DBMS_OUTPUT.PUT_LINE('datos de envio por farmacia actualizados con exito');
 27  EXCEPTION
 28   WHEN NO_DATA_FOUND THEN
 29      DBMS_OUTPUT.PUT_LINE('ERROR, NO SE ENCONTRÓ INFORMACIÓN');
 30    WHEN OTHERS THEN
 31      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR EN RECIBE_DRON_CENTRAL');
 32  COMMIT;
 33  END actualizarControlEnvios;
 34  /

Procedure created.
```

```
SQL>
SQL>
SQL> CREATE OR REPLACE TRIGGER t_expide_dron_central
  2  AFTER INSERT ON ExpideDronCentral
  3  FOR EACH ROW
  4  BEGIN
  5  INSERT INTO ControlMovimiento (
  6    cod_control_movimiento,
  7    dron,
  8    central_salida,
  9    fecha_salida
 10    ) VALUES (
 11      cod_control_mov_seq.NEXTVAL,
 12      :new.dron,
 13      :new.central,
 14      :new.fecha_salida_central
 15    );
 16    EXCEPTION
 17    WHEN NO_DATA_FOUND THEN
 18      DBMS_OUTPUT.PUT_LINE('no se encontraron valores en el trigger t_expide_dron_central');
 19    WHEN DUP_VAL_ON_INDEX THEN
 20      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de control_movimientos');
 21    WHEN OTHERS THEN
 22      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CONTROL_MOVIMIENTO');
 23    END t_expide_dron_central;
 24  /

Trigger created.

SQL>
SQL> CREATE OR REPLACE TRIGGER t_recibe_dron_central
  2  AFTER INSERT ON RecibeDronCentral
  3  FOR EACH ROW
  4  BEGIN
  5  UPDATE ControlMovimiento
  6      SET
  7          central_llegada = :new.central,
  8          fecha_llegada = :new.fecha_llegada_central
  9      WHERE dron = :new.dron AND central_llegada IS NULL AND fecha_llegada IS NULL;
 10  EXCEPTION
 11  WHEN NO_DATA_FOUND THEN
 12      DBMS_OUTPUT.PUT_LINE('no se encontraron valores en el trigger t_recibe_dron_central');
 13  END t_recibe_dron_central;
 14  /

Trigger created.
```

```
SQL>
SQL> CREATE OR REPLACE TRIGGER t_recibe_dron_central
  2  AFTER INSERT ON RecibeDronCentral
  3  FOR EACH ROW
  4  BEGIN
  5  UPDATE ControlMovimiento
  6      SET
  7          central_llegada = :new.central,
  8          fecha_llegada = :new.fecha_llegada_central
  9      WHERE dron = :new.dron AND central_llegada IS NULL AND fecha_llegada IS NULL;
 10  EXCEPTION
 11  WHEN NO_DATA_FOUND THEN
 12      DBMS_OUTPUT.PUT_LINE('no se encontraron valores en el trigger t_recibe_dron_central');
 13  END t_recibe_dron_central;
 14  /

Trigger created.

SQL>
SQL> CREATE OR REPLACE TRIGGER t_insertar_control_envio
  2  AFTER INSERT ON Farmacia
  3  FOR EACH ROW
  4  BEGIN
  5  INSERT INTO ControlEnvio (
  6    cod_control_envio,
  7    cod_farmacia_ctrl
  8  ) VALUES (
  9    cod_control_envio_seq.NEXTVAL,
 10    :new.cod_farmacia
 11  );
 12  EXCEPTION
 13    WHEN DUP_VAL_ON_INDEX THEN
 14      DBMS_OUTPUT.PUT_LINE('ya existe ese codigo de control_envio');
 15    WHEN OTHERS THEN
 16      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CONTROL_ENVIO' || SQLERRM);
 17  END t_insertar_control_envio;
 18  /

Trigger created.

SQL>
SQL> CREATE OR REPLACE TRIGGER t_tiempoEntrega
  2    BEFORE UPDATE OF fecha_entrega ON Envio
  3    FOR EACH ROW
  4    BEGIN
  5     :NEW.tiempo_entrega := calcularTiempoEntrega(:NEW.fecha_envio, :NEW.fecha_entrega);
  6  EXCEPTION
  7   WHEN OTHERS THEN
  8      DBMS_OUTPUT.PUT_LINE('ERROR AL INSERTAR CONTROL_ENVIO' || SQLERRM);
  9  END t_tiempoEntrega;
 10  /

Trigger created.

SQL>
SQL>
```

```
SQL>
SQL> SET SERVEROUTPUT ON;
SQL> set linesize 3000;
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------------------
SQL> ------------------------------------------------              Bloques Anonimos                                  ------------------------
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------------------
SQL>
SQL>
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------------------
SQL> ------------------------------------              Bloque para insertar horarios                              ------------------------
SQL> ------------------------------------------------------------------------------------------------------------------------------------------------------------
SQL> DECLARE
  2      v_cod_horario Horarios.cod_horario%TYPE;
  3      v_hora_inicio Horarios.hora_inicio%TYPE;
  4      v_hora_final Horarios.hora_final%TYPE;
  5      v_tipo_turno Horarios.tipo_turno%TYPE;
  6  BEGIN
  7      v_cod_horario := 'H01';
  8      v_hora_inicio := TO_DATE('5:00', 'HH24:MI:SS');
  9      v_hora_final := TO_DATE('13:00', 'HH24:MI:SS');
 10      v_tipo_turno := 'Matutino';
 11
 12      insertarHorario(v_cod_horario, v_hora_inicio, v_hora_final, v_tipo_turno);
 13
 14      v_cod_horario := 'H02';
 15      v_hora_inicio := TO_DATE('13:00', 'HH24:MI:SS');
 16      v_hora_final := TO_DATE('21:00', 'HH24:MI:SS');
 17      v_tipo_turno := 'Vespertino';
 18
 19      insertarHorario(v_cod_horario, v_hora_inicio, v_hora_final, v_tipo_turno);
 20
 21
 22      v_cod_horario := 'H03';
 23      v_hora_inicio := TO_DATE('21:00', 'HH24:MI:SS');
 24      v_hora_final := TO_DATE('5:00', 'HH24:MI:SS');
 25      v_tipo_turno := 'Nocturno';
 26
 27      insertarHorario(v_cod_horario, v_hora_inicio, v_hora_final, v_tipo_turno);
 28
 29  EXCEPTION
 30      WHEN DUP_VAL_ON_INDEX THEN
 31          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el horario: PK EXISTENTE');
 32      WHEN OTHERS THEN
 33          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el horario.' || SQLERRM);
 34  END;
 35  /
Horario Insertado existosamente
Horario Insertado existosamente
Horario Insertado existosamente

PL/SQL procedure successfully completed.

SQL>
```

```
SQL> DECLARE
  2      v_nombre Central.nombre%TYPE;
  3      v_latitud_central Central.latitud_central%TYPE;
  4      v_longitud_central Central.longitud_central%TYPE;
  5  BEGIN
  6
  7  -- codigo central 1
  8      v_nombre := 'Obarrio';
  9      v_latitud_central := '8.987362';
 10      v_longitud_central := '-79.520006';
 11
 12      insertarCentral(v_nombre, v_latitud_central, v_longitud_central);
 13
 14  -- codigo central 2
 15
 16      v_nombre := 'Condado del Rey';
 17      v_latitud_central := '9.032362';
 18      v_longitud_central := '-79.523426';
 19
 20      insertarCentral(v_nombre, v_latitud_central, v_longitud_central);
 21
 22  -- codigo central 3
 23
 24      v_nombre := 'San fransico';
 25      v_latitud_central := '8.988345';
 26      v_longitud_central := '-79.506384';
 27
 28      insertarCentral(v_nombre, v_latitud_central, v_longitud_central);
 29
 30      -- codigo central 4
 31
 32      v_nombre := 'El dorado';
 33      v_latitud_central := '9.008742';
 34      v_longitud_central := '79.534653';
 35
 36      insertarCentral(v_nombre, v_latitud_central, v_longitud_central);
 37
 38
 39  EXCEPTION
 40      WHEN DUP_VAL_ON_INDEX THEN
 41          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar la central: PK EXISTENTE');
 42      WHEN OTHERS THEN
 43          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar la central.' || SQLERRM);
 44  END;
 45  /
central Insertada existosamente
central Insertada existosamente
central Insertada existosamente
central Insertada existosamente

PL/SQL procedure successfully completed.
```

```
SQL> DECLARE
  2      v_nombre Personal.nombre%TYPE;
  3      v_apellido Personal.apellido%TYPE;
  4      v_cedula Personal.cedula%TYPE;
  5      v_fecha_nacimiento Personal.fecha_nacimiento%TYPE;
  6      v_sexo Personal.sexo%TYPE;
  7      v_cod_horario_personal Personal.cod_horario_personal%TYPE;
  8      v_central_nombre Central.nombre%TYPE;
  9      v_telefono_mecanico Telefono_Mecanico.telefono_mecanico%TYPE;
 10  BEGIN
 11
 12      v_nombre := 'Ana'; -- codigo de personal 1
 13      v_apellido := 'Garcia';
 14      v_cedula := '8-888-8888';
 15      v_fecha_nacimiento := TO_DATE('02/03/2003','DD/MM/YYYY');
 16      v_sexo := 'F';
 17      v_cod_horario_personal := 'H01';
 18      v_central_nombre := 'Obarrio';
 19      v_telefono_mecanico := '61123456';
 20
 21      insertarMecanico(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_central_nombre, v_telefono_mecanico);
 22
 23      v_nombre := 'Pedro';-- codigo de personal 2
 24      v_apellido := 'Rodriguez';
 25      v_cedula := '4-444-4444';
 26      v_fecha_nacimiento := TO_DATE('10/09/1985', 'DD/MM/YYYY');
 27      v_sexo := 'M';
 28      v_cod_horario_personal := 'H02';
 29      v_central_nombre := 'Condado del Rey';
 30      v_telefono_mecanico := '62234567';
 31
 32      insertarMecanico(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_central_nombre, v_telefono_mecanico);
 33
 34
 35      v_nombre := 'María';-- codigo de personal 3
 36      v_apellido := 'Martínez';
 37      v_cedula := '1-111-1111';
 38      v_fecha_nacimiento := TO_DATE('15/11/1998', 'DD/MM/YYYY');
 39      v_sexo := 'F';
 40      v_cod_horario_personal := 'H03';
 41      v_central_nombre := 'San fransico';
 42      v_telefono_mecanico := '63345678';
 43
 44      insertarMecanico(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_central_nombre, v_telefono_mecanico);
 45
 46
 47      v_nombre := 'Carlos';-- codigo de personal 4
 48      v_apellido := 'Pérez';
 49      v_cedula := '5-555-5555';
 50      v_fecha_nacimiento := TO_DATE('20/03/1992', 'DD/MM/YYYY');
 51      v_sexo := 'M';
 52      v_cod_horario_personal := 'H01';
 53      v_central_nombre := 'El dorado';
 54      v_telefono_mecanico := '64456789';
 55
```

```
 55
 56      insertarMecanico(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_central_nombre, v_telefono_mecanico);
 57
 58
 59      v_nombre := 'Luisa';-- codigo de personal 5
 60      v_apellido := 'Gómez';
 61      v_cedula := '7-777-7777';
 62      v_fecha_nacimiento := TO_DATE('25/07/1989', 'DD/MM/YYYY');
 63      v_sexo := 'F';
 64      v_cod_horario_personal := 'H02';
 65      v_central_nombre := 'Obarrio';
 66      v_telefono_mecanico := '65567890';
 67
 68      insertarMecanico(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_central_nombre, v_telefono_mecanico);
 69
 70  EXCEPTION
 71      WHEN DUP_VAL_ON_INDEX THEN
 72          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el mecanico: PK EXISTENTE');
 73      WHEN OTHERS THEN
 74          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el mecanico.' || SQLERRM);
 75  END;
 76  /
Mecanico Insertado existosamente
Mecanico Insertado existosamente
Mecanico Insertado existosamente
Mecanico Insertado existosamente
Mecanico Insertado existosamente

PL/SQL procedure successfully completed.
```

```
SQL>
SQL> DECLARE
  2      v_nombre Personal.nombre%TYPE;
  3      v_apellido Personal.apellido%TYPE;
  4      v_cedula Personal.cedula%TYPE;
  5      v_fecha_nacimiento Personal.fecha_nacimiento%TYPE;
  6      v_sexo Personal.sexo%TYPE;
  7      v_cod_horario_personal Personal.cod_horario_personal%TYPE;
  8      v_telefono_supervisor Telefono_Supervisor.telefono_supervisor%TYPE;
  9  BEGIN
 10      v_nombre := 'Jorge';-- codigo de personal 6
 11      v_apellido := 'Castro';
 12      v_cedula := '8-972-1616';
 13      v_fecha_nacimiento := TO_DATE('12/06/1980', 'DD/MM/YYYY');
 14      v_sexo := 'M';
 15      v_cod_horario_personal := 'H02';
 16      v_telefono_supervisor := '68563478';
 17      insertarSupervisor(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_telefono_supervisor);
 18
 19      v_nombre := 'Sofía';-- codigo de personal 7
 20      v_apellido := 'López';
 21      v_cedula := '8-972-1617';
 22      v_fecha_nacimiento := TO_DATE('23/09/1975', 'DD/MM/YYYY');
 23      v_sexo := 'F';
 24      v_cod_horario_personal := 'H03';
 25      v_telefono_supervisor := '68563479';
 26      insertarSupervisor(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_telefono_supervisor);
 27
 28      v_nombre := 'Eduardo';-- codigo de personal 8
 29      v_apellido := 'Gutiérrez';
 30      v_cedula := '8-972-1618';
 31      v_fecha_nacimiento := TO_DATE('15/02/1988', 'DD/MM/YYYY');
 32      v_sexo := 'M';
 33      v_cod_horario_personal := 'H01';
 34      v_telefono_supervisor := '68563480';
 35      insertarSupervisor(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_telefono_supervisor);
 36
 37      v_nombre := 'Laura';-- codigo de personal 9
 38      v_apellido := 'Pérez';
 39      v_cedula := '8-972-1619';
 40      v_fecha_nacimiento := TO_DATE('20/10/1990', 'DD/MM/YYYY');
 41      v_sexo := 'F';
 42      v_cod_horario_personal := 'H02';
 43      v_telefono_supervisor := '68563481';
 44
 45      insertarSupervisor(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_telefono_supervisor);
 46
 47      v_nombre := 'Manuel';-- codigo de personal 10
 48      v_apellido := 'Ramírez';
 49      v_cedula := '8-972-1620';
 50      v_fecha_nacimiento := TO_DATE('05/03/1987', 'DD/MM/YYYY');
 51      v_sexo := 'M';
 52      v_cod_horario_personal := 'H03';
 53      v_telefono_supervisor := '68563482';
 54
 55      insertarSupervisor(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_telefono_supervisor);
```

```
 55      insertarSupervisor(v_nombre, v_apellido, v_cedula, v_fecha_nacimiento, v_sexo, v_cod_horario_personal, v_telefono_supervisor);
 56      EXCEPTION
 57      WHEN DUP_VAL_ON_INDEX THEN
 58          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el supervisor: PK EXISTENTE');
 59      WHEN OTHERS THEN
 60          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el supervisor.' || SQLERRM);
 61  END;
 62  /
Supervisor Insertado existosamente
Supervisor Insertado existosamente
Supervisor Insertado existosamente
Supervisor Insertado existosamente
Supervisor Insertado existosamente

PL/SQL procedure successfully completed.
```

```
SQL>
SQL> DECLARE
  2      v_ruc Farmacia.ruc%TYPE;
  3      v_nombre Farmacia.nombre%TYPE;
  4      v_correo Farmacia.correo%TYPE;
  5      v_longitud_farmacia Farmacia.longitud_farmacia%TYPE;
  6      v_latitud_farmacia Farmacia.latitud_farmacia%TYPE;
  7      v_telefono_farmacia Farmacia_Telefono.telefono_farmacia%TYPE;
  8  BEGIN
  9       v_ruc := '123-456-789012-3';
 10      v_nombre := 'Farmacias Arrocha';
 11      v_correo := 'farrochaobarrio@arrocha.com'; --  <------- esta farmacia es de obarrio con codigo 1
 12      v_latitud_farmacia := 8.986067;
 13      v_longitud_farmacia := -79.517493;
 14      v_telefono_farmacia := '360-4391';
 15
 16      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 17
 18       v_ruc := '123-436-742312-3';
 19      v_nombre := 'Farma Value';
 20      v_correo := 'farmavaluecangrejo@farmavalue.com'; --  <------- esta farmacia es del cangrejo con codigo 2
 21      v_latitud_farmacia := 8.989840;
 22      v_longitud_farmacia := -79.524381;
 23      v_telefono_farmacia := '302-3830';
 24
 25      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 26
 27      v_ruc := '411-191-912411-1';
 28      v_nombre := 'Metro Plus';
 29      v_correo := 'metroplussf@metrops.com';--  <------- esta farmacia es de san fransisco con codigo 3
 30      v_latitud_farmacia := 8.989556;
 31      v_longitud_farmacia := -79.504941;
 32      v_telefono_farmacia := '226-6527';
 33
 34      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 35
 36       v_ruc := '793-486-769312-3';
 37      v_nombre := 'Farmacias El Javillo';
 38      v_correo := 'eljavilloviaisrrael@javillo.com';--  <------- esta farmacia es de via isrrael con codigo 4
 39      v_latitud_farmacia := 8.987670;
 40      v_longitud_farmacia := -79.505263;
 41      v_telefono_farmacia := '301-4019';
 42
 43      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 44
 45      v_ruc := '793-486-769312-3';
 46      v_nombre := 'Farmaias El Javillo';
 47      v_correo := 'eljavilloviaporras@javillo.com';--  <------- esta farmacia es de via porras con codigo 5
 48      v_latitud_farmacia := 8.996377;
 49      v_longitud_farmacia := -79.510362;
 50      v_telefono_farmacia := '301-4087';
 51
 52      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 53
```

```
 52      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 53
 54      v_ruc := '411-191-912411-1';
 55      v_nombre := 'Metro Plus';
 56      v_correo := 'metroplusdorado@metrop.com';--  <------- esta farmacia es del dorado con codigo 6
 57      v_latitud_farmacia := 9.006436;
 58      v_longitud_farmacia := -79.5339301;
 59      v_telefono_farmacia := '395-3491';
 60
 61      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 62
 63      v_ruc := '123-456-789012-3';
 64      v_nombre := 'Farmacias Arrocha';
 65      v_correo := 'arrocha5mayo@metrop.com';--  <------- esta farmacia es de la 5 de mayo con codigo 7
 66      v_latitud_farmacia := 8.959030;
 67      v_longitud_farmacia := -79.541246;
 68      v_telefono_farmacia := '395-3433';
 69
 70
 71      insertarFarmacia(v_ruc, v_nombre, v_correo, v_longitud_farmacia, v_latitud_farmacia, v_telefono_farmacia);
 72
 73  EXCEPTION
 74      WHEN DUP_VAL_ON_INDEX THEN
 75          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar la farmacia: PK EXISTENTE');
 76      WHEN OTHERS THEN
 77          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar la farmacia.' || SQLERRM);
 78  END;
 79  /
Farmacia Insertada existosamente
Farmacia Insertada existosamente
Farmacia Insertada existosamente
Farmacia Insertada existosamente
Farmacia Insertada existosamente
Farmacia Insertada existosamente
Farmacia Insertada existosamente

PL/SQL procedure successfully completed.

SQL>
```

```
SQL>
SQL> DECLARE
  2      v_marca Dron.marca%TYPE;
  3      v_modelo Dron.modelo%TYPE;
  4      v_fecha_fabricacion Dron.fecha_fabricacion%TYPE;
  5      v_disponibilidad Dron.disponibilidad%TYPE;
  6      v_capacidad_carga Dron.capacidad_carga%TYPE;
  7      v_velocidad_promedio Dron.velocidad_promedio%TYPE;
  8  BEGIN
  9  --codigo de dron 1
 10      v_marca := 'DJI';
 11      v_modelo := 'Phantom 4 Pro';
 12      v_fecha_fabricacion := TO_DATE('01/01/2023', 'DD/MM/YYYY');
 13      v_disponibilidad := 'Y';
 14      v_capacidad_carga := 25;
 15      v_velocidad_promedio := 70;
 16
 17      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
 18
 19  --codigo de dron 2
 20
 21      v_marca := 'Parrot';
 22      v_modelo := 'Anafi';
 23      v_fecha_fabricacion := TO_DATE('01/01/2023', 'DD/MM/YYYY');
 24      v_disponibilidad := 'Y';
 25      v_capacidad_carga := 30;
 26      v_velocidad_promedio := 65;
 27
 28      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
 29
 30  --codigo de dron 3
 31
 32      v_marca := 'Autel Robotics';
 33      v_modelo := 'EVO II Pro';
 34      v_fecha_fabricacion := TO_DATE('02/01/2023', 'DD/MM/YYYY');
 35      v_disponibilidad := 'Y';
 36      v_capacidad_carga := 30;
 37      v_velocidad_promedio := 72;
 38
 39      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
 40
 41  --codigo de dron 4
 42
 43      v_marca := 'Yuneec';
 44      v_modelo := 'Typhoon H Plus';
 45      v_fecha_fabricacion := TO_DATE('02/01/2023', 'DD/MM/YYYY');
 46      v_disponibilidad := 'Y';
 47      v_capacidad_carga := 28;
 48      v_velocidad_promedio := 68;
 49
 50      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
 51
```

```
51
52  --codigo de dron 5
53
54      v_marca := 'Skydio';
55      v_modelo := 'X2';
56      v_fecha_fabricacion := TO_DATE('02/01/2023', 'DD/MM/YYYY');
57      v_disponibilidad := 'Y';
58      v_capacidad_carga := 20;
59      v_velocidad_promedio := 75;
60
61      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
62
63  --codigo de dron 6
64
65      v_marca := 'Ehang';
66      v_modelo := 'Ghostdrone 2.0 VR';
67      v_fecha_fabricacion := TO_DATE('05/01/2023', 'DD/MM/YYYY');
68      v_disponibilidad := 'Y';
69      v_capacidad_carga := 22;
70      v_velocidad_promedio := 68;
71
72      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
73
74  --codigo de dron 7
75
76      v_marca := 'Hubsan';
77      v_modelo := 'H501S X4';
78      v_fecha_fabricacion := TO_DATE('05/01/2023', 'DD/MM/YYYY');
79      v_disponibilidad := 'Y';
80      v_capacidad_carga := 28;
81      v_velocidad_promedio := 60;
82
83      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
84
85  --codigo de dron 8
86
87      v_marca := 'Walkera';
88      v_modelo := 'Voyager 4';
89      v_fecha_fabricacion := TO_DATE('05/01/2023', 'DD/MM/YYYY');
90      v_disponibilidad := 'Y';
91      v_capacidad_carga := 26;
92      v_velocidad_promedio := 71;
93
94      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
95
```

```
 95
 96  --codigo de dron 9
 97
 98      v_marca := 'PowerVision';
 99      v_modelo := 'PowerEye';
100      v_fecha_fabricacion := TO_DATE('06/01/2023', 'DD/MM/YYYY');
101      v_disponibilidad := 'Y';
102      v_capacidad_carga := 32;
103      v_velocidad_promedio := 67;
104
105      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
106
107  --codigo de dron 10
108
109      v_marca := 'Holy Stone';
110      v_modelo := 'HS100';
111      v_fecha_fabricacion := TO_DATE('06/01/2023', 'DD/MM/YYYY');
112      v_disponibilidad := 'Y';
113      v_capacidad_carga := 21;
114      v_velocidad_promedio := 63;
115
116      insertarDron(v_marca, v_modelo, v_fecha_fabricacion, v_disponibilidad, v_capacidad_carga, v_velocidad_promedio);
117
118  EXCEPTION
119      WHEN DUP_VAL_ON_INDEX THEN
120          DBMS_OUTPUT.PUT_LINE('Error al insertar el dron en el intento : PK EXISTENTE');
121      WHEN OTHERS THEN
122          DBMS_OUTPUT.PUT_LINE('Error en el bloque anónimo: ' || SQLERRM);
123      COMMIT;
124  END;
125  /
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente
Dron Insertado existosamente

PL/SQL procedure successfully completed.

SQL>
```

```
SQL>
SQL> DECLARE
  2      v_peso Envio.peso%TYPE;
  3      v_latitud_final Envio.latitud_final %TYPE;
  4      v_longitud_final Envio.longitud_final %TYPE;
  5      v_cod_farmacia_envio Farmacia.cod_farmacia%TYPE;
  6      v_cod_supervisor_envio Supervisor.cod_supervisor%TYPE;
  7      v_cod_dron_envio Dron.cod_dron%TYPE;
  8  BEGIN
  9  --codigo de envio 1
 10      v_peso := 8; --kg
 11      v_latitud_final := 9.006621;
 12      v_longitud_final := -79.507535;
 13      v_cod_farmacia_envio := 5; --via porras
 14      v_cod_supervisor_envio := 10;
 15      v_cod_dron_envio := 1;
 16      InsertarEnvio(v_peso, v_latitud_final,v_longitud_final, v_cod_farmacia_envio, v_cod_supervisor_envio, v_cod_dron_envio);
 17
 18      --codigo de envio 2
 19      v_peso := 2; --kg
 20      v_latitud_final := 8.994058;
 21      v_longitud_final := -79.512674;
 22      v_cod_farmacia_envio := 3; --san francisco
 23      v_cod_supervisor_envio := 6;
 24      v_cod_dron_envio := 2;
 25      InsertarEnvio(v_peso, v_latitud_final,v_longitud_final, v_cod_farmacia_envio, v_cod_supervisor_envio, v_cod_dron_envio);
 26
 27      --codigo de envio 3
 28      v_peso := 20; --kg
 29      v_latitud_final := 8.972314;
 30      v_longitud_final := -79.535070;
 31      v_cod_farmacia_envio := 5; --via porras
 32      v_cod_supervisor_envio := 7;
 33      v_cod_dron_envio := 3;
 34      InsertarEnvio(v_peso, v_latitud_final,v_longitud_final, v_cod_farmacia_envio, v_cod_supervisor_envio, v_cod_dron_envio);
 35
 36      --codigo de envio 4
 37      v_peso := 8; --kg
 38      v_latitud_final := 8.958054;
 39      v_longitud_final := -79.549735;
 40      v_cod_farmacia_envio := 7; -- 5 de mayo
 41      v_cod_supervisor_envio := 8;
 42      v_cod_dron_envio := 4;
 43      InsertarEnvio(v_peso, v_latitud_final,v_longitud_final, v_cod_farmacia_envio, v_cod_supervisor_envio, v_cod_dron_envio);
 44
 45      --codigo de envio 5
 46      v_peso := 7; --kg
 47      v_latitud_final := 8.994674;
 48      v_longitud_final := -79.531869;
 49      v_cod_farmacia_envio := 2; -- el cangrejo
 50      v_cod_supervisor_envio := 9;
 51      v_cod_dron_envio := 5;
 52      InsertarEnvio(v_peso, v_latitud_final,v_longitud_final, v_cod_farmacia_envio, v_cod_supervisor_envio, v_cod_dron_envio);
```

```
53
54      --codigo de envio 6
55      v_peso := 15; --kg
56      v_latitud_final := 9.008344;
57      v_longitud_final := -79.538909;
58      v_cod_farmacia_envio := 6;-- el dorado
59      v_cod_supervisor_envio := 10;
60      v_cod_dron_envio := 6;
61      InsertarEnvio(v_peso, v_latitud_final,v_longitud_final, v_cod_farmacia_envio, v_cod_supervisor_envio, v_cod_dron_envio);
62
63  EXCEPTION
64      WHEN DUP_VAL_ON_INDEX THEN
65          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el envio: PK EXISTENTE');
66      WHEN OTHERS THEN
67          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar el envio.' || SQLERRM);
68  END;
69  /
envio Insertado existosamente
envio Insertado existosamente
envio Insertado existosamente
envio Insertado existosamente
envio Insertado existosamente
envio Insertado existosamente

PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL> ------------------------------------------------------------------------------------------------------------------------
SQL> ----------------------          Bloque para insertar cuando se expide un dron          ----------------------
SQL> ------------------------------------------------------------------------------------------------------------------------
SQL>
SQL> DECLARE
  2      v_dron Dron.cod_dron%TYPE;
  3      v_central Central.cod_central%TYPE;
  4  BEGIN
  5      v_dron := 1;
  6      v_central := 1;
  7      expedirDron(
  8          p_dron => v_dron,
  9          p_central => v_central
 10      );
 11
 12          v_dron := 2;
 13      v_central := 2;
 14      expedirDron(
 15          p_dron => v_dron,
 16          p_central => v_central
 17      );
 18
 19          v_dron := 3;
 20      v_central := 3;
 21      expedirDron(
 22          p_dron => v_dron,
 23          p_central => v_central
 24      );
 25
 26          v_dron := 4;
 27      v_central := 1;
 28      expedirDron(
 29          p_dron => v_dron,
 30          p_central => v_central
 31      );
 32          v_dron := 5;
 33      v_central := 3;
 34      expedirDron(
 35          p_dron => v_dron,
 36          p_central => v_central
 37      );
```

```
36          p_central => v_central
37      );
38
39          v_dron := 6;
40      v_central := 4;
41      expedirDron(
42          p_dron => v_dron,
43          p_central => v_central
44      );
45  EXCEPTION
46      WHEN DUP_VAL_ON_INDEX THEN
47          DBMS_OUTPUT.PUT_LINE('bloque anonimo, ya existe ese codigo de ExpideDronCentral');
48      WHEN OTHERS THEN
49          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo la salida de un dron de una central.');
50  END;
51  /
registro de salida de dron de la central 1 insertado correctamente
registro de salida de dron de la central 2 insertado correctamente
registro de salida de dron de la central 3 insertado correctamente
registro de salida de dron de la central 1 insertado correctamente
registro de salida de dron de la central 3 insertado correctamente
registro de salida de dron de la central 4 insertado correctamente

PL/SQL procedure successfully completed.
```

```
SQL>
SQL> DECLARE
  2      v_fecha_envio Envio.fecha_envio%TYPE;
  3      v_cod_envio Envio.cod_envio%TYPE;
  4  BEGIN
  5      v_cod_envio := 1;
  6      llegaFarmacia(
  7          p_cod_envio => v_cod_envio
  8      );
  9          v_cod_envio := 2;
 10      llegaFarmacia(
 11          p_cod_envio => v_cod_envio
 12      );
 13          v_cod_envio := 3;
 14      llegaFarmacia(
 15          p_cod_envio => v_cod_envio
 16      );
 17          v_cod_envio := 4;
 18      llegaFarmacia(
 19          p_cod_envio => v_cod_envio
 20      );
 21          v_cod_envio := 5;
 22      llegaFarmacia(
 23          p_cod_envio => v_cod_envio
 24      );
 25          v_cod_envio := 6;
 26      llegaFarmacia(
 27          p_cod_envio => v_cod_envio
 28      );
 29  EXCEPTION
 30      WHEN NO_DATA_FOUND THEN
 31          DBMS_OUTPUT.PUT_LINE('No se encontro un envio en la actualizacion');
 32      WHEN OTHERS THEN
 33          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar cuando un dron llega a la farmacia.');
 34  END;
 35  /
Fecha de llegada a la farmacia actualizada exitosamente
Fecha de llegada a la farmacia actualizada exitosamente
Fecha de llegada a la farmacia actualizada exitosamente
Fecha de llegada a la farmacia actualizada exitosamente
Fecha de llegada a la farmacia actualizada exitosamente
Fecha de llegada a la farmacia actualizada exitosamente

PL/SQL procedure successfully completed.
```

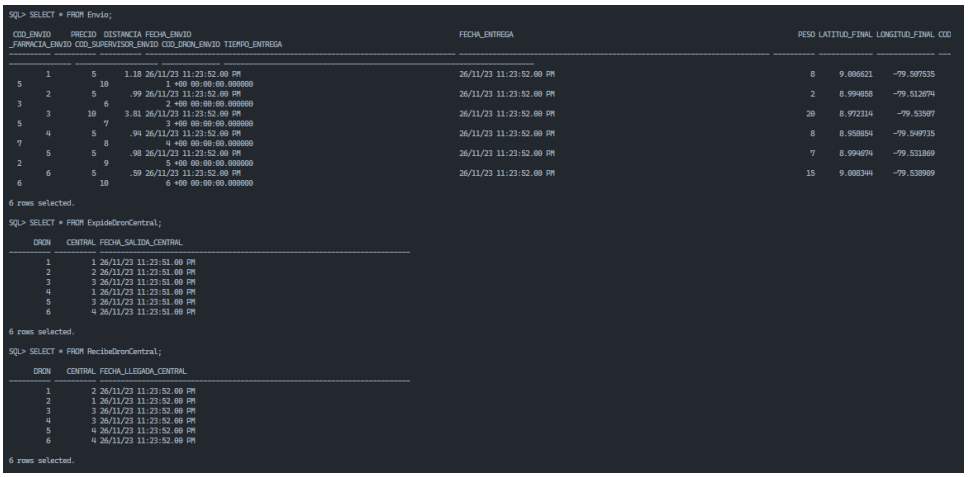
```
SQL>
SQL> DECLARE
  2      v_cod_envio Envio.cod_envio%TYPE;
  3  BEGIN
  4      v_cod_envio := 1;
  5      entregaPedido(
  6          p_cod_envio => v_cod_envio
  7      );
  8      v_cod_envio := 2;
  9      entregaPedido(
 10          p_cod_envio => v_cod_envio
 11      );
 12      v_cod_envio := 3;
 13      entregaPedido(
 14          p_cod_envio => v_cod_envio
 15      );
 16      v_cod_envio := 4;
 17      entregaPedido(
 18          p_cod_envio => v_cod_envio
 19      );
 20      v_cod_envio := 5;
 21      entregaPedido(
 22          p_cod_envio => v_cod_envio
 23      );
 24      v_cod_envio := 6;
 25      entregaPedido(
 26          p_cod_envio => v_cod_envio
 27      );
 28  EXCEPTION
 29      WHEN NO_DATA_FOUND THEN
 30          DBMS_OUTPUT.PUT_LINE('No se encontro un envio en la actualizacion');
 31      WHEN OTHERS THEN
 32          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al insertar la entrega de un envio.');
 33  END;
 34  /
Fecha de entrega al cliente actualizada exitosamente
Fecha de entrega al cliente actualizada exitosamente
Fecha de entrega al cliente actualizada exitosamente
Fecha de entrega al cliente actualizada exitosamente
Fecha de entrega al cliente actualizada exitosamente
Fecha de entrega al cliente actualizada exitosamente

PL/SQL procedure successfully completed.

SQL>
```

```
SQL>
SQL> DECLARE
  2      v_dron Dron.cod_dron%TYPE;
  3      v_central Central.cod_central%TYPE;
  4  BEGIN
  5      v_dron := 1;
  6      v_central := 2;
  7      recibirDron(
  8          p_dron => v_dron,
  9          p_central => v_central
 10      );
 11      v_dron := 2;
 12      v_central := 1;
 13      recibirDron(
 14          p_dron => v_dron,
 15          p_central => v_central
 16      );
 17      v_dron := 3;
 18      v_central := 3;
 19      recibirDron(
 20          p_dron => v_dron,
 21          p_central => v_central
 22      );
 23      v_dron := 4;
 24      v_central := 3;
 25      recibirDron(
 26          p_dron => v_dron,
 27          p_central => v_central
 28      );
 29      v_dron := 5;
 30      v_central := 4;
 31      recibirDron(
 32          p_dron => v_dron,
 33          p_central => v_central
 34      );
 35      v_dron := 6;
 36      v_central := 4;
 37      recibirDron(
 38          p_dron => v_dron,
 39          p_central => v_central
 40      );
 41  EXCEPTION
 42      WHEN DUP_VAL_ON_INDEX THEN
 43          DBMS_OUTPUT.PUT_LINE('bloque anonimo, ya existe ese codigo de RecibeDronCentral');
 44      WHEN OTHERS THEN
 45          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al llegar un dron a una central.');
 46  END;
 47  /
registro de llegada de dron de la central 2 insertado correctamente
registro de llegada de dron de la central 1 insertado correctamente
registro de llegada de dron de la central 3 insertado correctamente
registro de llegada de dron de la central 3 insertado correctamente
registro de llegada de dron de la central 4 insertado correctamente
registro de llegada de dron de la central 4 insertado correctamente

PL/SQL procedure successfully completed.
```

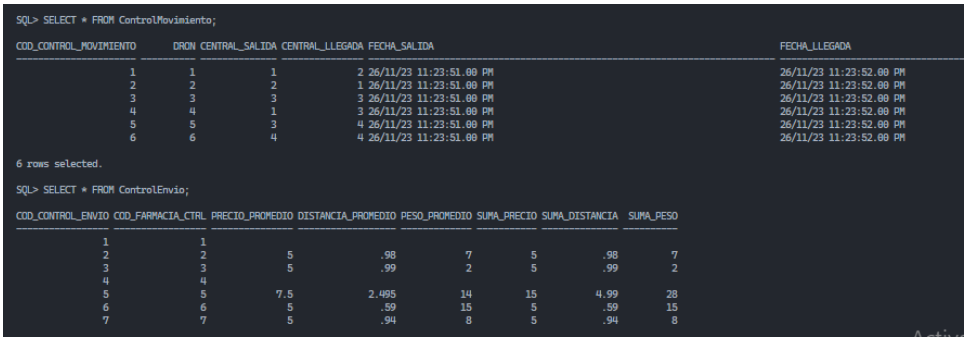
```
SQL> ------------------------------------------------------------------------
SQL> ------------------------control_envios-----------------------------
SQL> ------------------------------------------------------------------------
SQL> BEGIN
  2  actualizarControlEnvios();
  3  EXCEPTION
  4      WHEN NO_DATA_FOUND THEN
  5          DBMS_OUTPUT.PUT_LINE('bloque anonimo, No se encontro un registro en la tabla contrl_envio');
  6      WHEN OTHERS THEN
  7          DBMS_OUTPUT.PUT_LINE('Error en el bloque anonimo al actualizar los datos de la tabla control_envios.');
  8  END;
  9  /
datos de envio por farmacia actualizados con exito

PL/SQL procedure successfully completed.
```

```
SQL> SELECT * FROM Horarios;

COD_ HORA_INI HORA_FIN TIPO_TURNO
---- -------- -------- ----------
H01  01/11/23 01/11/23 Matutino
H02  01/11/23 01/11/23 Vespertino
H03  01/11/23 01/11/23 Nocturno

SQL> SELECT * FROM Central;

COD_CENTRAL NOMBRE                LONGITUD_CENTRAL                                 LATITUD_CENTRAL
----------- --------------------- ------------------------------------------------ ------------------------------------------------
          1 Obarrio               8.987362                                         -79.520086
          2 Condado del Rey       9.032362                                         -79.523426
          3 San fransico          8.988345                                         -79.506384
          4 El dorado             9.008742                                         79.534653

SQL> SELECT * FROM Personal;

COD_PERSONAL NOMBRE                                             APELLIDO                                          CEDULA     FECHA_NA S COD_
------------ -------------------------------------------------- ------------------------------------------------- ---------- -------- - -----
           1 Ana                                                Garcia                                            8-888-8888 02/03/03 F H01
           2 Pedro                                              Rodriguez                                         4-4444-4444 10/09/85 M H02
           3 Maria                                              Martinez                                          1-111-1111 15/11/98 F H03
           4 Carlos                                             Pérez                                             5-555-5555 20/03/92 M H01
           5 Luisa                                              Gomez                                             7-777-7777 25/07/89 F H02
           6 Jorge                                              Castro                                            8-972-1616 12/06/80 M H02
           7 Sofia                                              López                                             8-972-1617 23/09/75 F H03
           8 Eduardo                                            Gutiérrez                                         8-972-1618 15/02/88 M H01
           9 Laura                                              Pérez                                             8-972-1619 20/10/90 F H02
          10 Manuel                                             Ramirez                                           8-972-1620 05/03/87 M H03

10 rows selected.

SQL> SELECT * FROM Mecanico m;

COD_MECANICO COD_CENTRAL_MECANICO
------------ --------------------
           1                    1
           2                    2
           3                    3
           4                    4
           5                    1

SQL> SELECT * FROM Supervisor;

COD_SUPERVISOR
--------------
             6
             7
             8
             9
            10
```

```
COD_MECANICO_TEL TELEFONO_MECANICO
---------------- ------------------
               1 61123456
               2 62234567
               3 63345678
               4 64456789
               5 65567890

SQL> SELECT * FROM Telefono_Supervisor;

COD_SUPERVISOR_TEL TELEFONO_SUPERVISOR
------------------ -------------------
                 6 68563478
                 7 68563479
                 8 68563480
                 9 68563481
                10 68563482

SQL> SELECT f.*, t.telefono_farmacia FROM Farmacia f join Farmacia_Telefono t on f.cod_farmacia = t.cod_farmacia_tel;

COD_FARMACIA RUC                                               NOMBRE                    CORREO                                                 LONGITUD_FARMACIA LATITUD_FARMACIA TELEFONO_FARMACIA
------------ ------------------------------------------------- ------------------------- ------------------------------------------------------ ----------------- ---------------- ------------------------------
           1 123-456-789012-3                                  Farmacias Arrocha         farrochaobarrio@arrocha.com                                  -79.517493       8.986007 360-4391
           2 123-436-742312-3                                  Farma Value               farmavaluecangrejo@farmavalue.com                            -79.524381       8.98884 302-3830
           3 411-191-912411-1                                  Metro Plus                metroplussf@metrops.com                                      -79.504941       8.989556 226-6527
           4 793-486-769312-3                                  Farmacias El Javillo      eljavilloviaisrrael@javillo.com                              -79.505263        8.98767 301-4019
           5 793-486-769312-3                                  Farmacias El Javillo      eljavilloviaporras@javillo.com                               -79.510362       8.996377 301-4887
           6 411-191-912411-1                                  Metro Plus                metroplusdorado@metrop.com                                   -79.53393        9.006936 395-3491
           7 123-456-789012-3                                  Farmacias Arrocha         arrochaSmayo@metrop.com                                      -79.542246       8.95983 395-3433

7 rows selected.

SQL> --SELECT * FROM Farmacia_Telefono;
SQL> SELECT * FROM Dron;

  COD_DRON MARCA                                             MODELO                                            FECHA_FA D CAPACIDAD_CARGA VELOCIDAD_PROMEDIO
---------- ------------------------------------------------- ------------------------------------------------- -------- - -------------- ----------------
         1 DJI                                               Phantom 4 Pro                                     01/01/23 Y             25               70
         2 Parrot                                            Anafi                                             01/01/23 Y             30               65
         3 Autel Robotics                                    EVO II Pro                                        02/01/23 Y             30               72
         4 Yuneec                                            Typhoon H Plus                                    02/01/23 Y             28               68
         5 Skydio                                            X2                                                02/01/23 Y             20               75
         6 Ehang                                             Ghostdrone 2.0 VR                                 05/01/23 Y             22               68
         7 Hubsan                                            H501S X4                                          05/01/23 Y             28               69
         8 Walkera                                           Voyager 4                                         05/01/23 Y             26               71
         9 PowerVision                                       PowerEye                                          06/01/23 Y             32               67
        10 Holy Stone                                        HS100                                             06/01/23 Y             21               63

10 rows selected.
```



```
SQL> SELECT * FROM Envio;

 COD_ENVIO     PRECIO  DISTANCIA FECHA_ENVIO
_FARMACIA_ENVIO COD_SUPERVISOR_ENVIO COD_DRON_ENVIO TIEMPO_ENTREGA
---------- --------- --------- ---------------------------------------------------------------------------
FECHA_ENTREGA                                                                         PESO LATITUD_FINAL LONGITUD_FINAL COD
--------------------------------------------------------------------------- ---------- ------------ ------------ ---
         1         5      1.18 26/11/23 11:23:52.00 PM
 5                 10                 1 +00 00:00:00.000000
26/11/23 11:23:52.00 PM                                                              8    9.006621    -79.597535

         2         5       .99 26/11/23 11:23:52.00 PM
 3                  6                 2 +00 00:00:00.000000
26/11/23 11:23:52.00 PM                                                              2    8.994858    -79.512674

         3        10      3.81 26/11/23 11:23:52.00 PM
 5                  7                 3 +00 00:00:00.000000
26/11/23 11:23:52.00 PM                                                             20    8.972314     -79.53597

         4         5       .94 26/11/23 11:23:52.00 PM
 7                  8                 4 +00 00:00:00.000000
26/11/23 11:23:52.00 PM                                                              8    8.958854    -79.549735

         5         5       .98 26/11/23 11:23:52.00 PM
 2                  9                 5 +00 00:00:00.000000
26/11/23 11:23:52.00 PM                                                              7    8.994674    -79.531869

         6         5       .59 26/11/23 11:23:52.00 PM
 6                 10                 6 +00 00:00:00.000000
26/11/23 11:23:52.00 PM                                                             15    9.008344    -79.538909

6 rows selected.

SQL> SELECT * FROM ExpideDronCentral;

      DRON    CENTRAL FECHA_SALIDA_CENTRAL
---------- ---------- ---------------------------------------------------------------------------
         1          1 26/11/23 11:23:51.00 PM
         2          2 26/11/23 11:23:51.00 PM
         3          3 26/11/23 11:23:51.00 PM
         4          1 26/11/23 11:23:51.00 PM
         5          3 26/11/23 11:23:51.00 PM
         6          4 26/11/23 11:23:51.00 PM

6 rows selected.

SQL> SELECT * FROM RecibeDronCentral;

      DRON    CENTRAL FECHA_LLEGADA_CENTRAL
---------- ---------- ---------------------------------------------------------------------------
         1          2 26/11/23 11:23:52.00 PM
         2          1 26/11/23 11:23:52.00 PM
         3          3 26/11/23 11:23:52.00 PM
         4          3 26/11/23 11:23:52.00 PM
         5          4 26/11/23 11:23:52.00 PM
         6          4 26/11/23 11:23:52.00 PM

6 rows selected.
```



```
SQL> SELECT * FROM ControlMovimiento;

COD_CONTROL_MOVIMIENTO       DRON CENTRAL_SALIDA CENTRAL_LLEGADA FECHA_SALIDA                                                                         FECHA_LLEGADA
---------------------- ---------- ------------- ---------------- ---------------------------------------------------------------------------
                     1          1             1                2 26/11/23 11:23:51.00 PM                                                             26/11/23 11:23:52.00 PM
                     2          2             2                1 26/11/23 11:23:51.00 PM                                                             26/11/23 11:23:52.00 PM
                     3          3             3                3 26/11/23 11:23:51.00 PM                                                             26/11/23 11:23:52.00 PM
                     4          4             1                3 26/11/23 11:23:51.00 PM                                                             26/11/23 11:23:52.00 PM
                     5          5             3                4 26/11/23 11:23:51.00 PM                                                             26/11/23 11:23:52.00 PM
                     6          6             4                4 26/11/23 11:23:51.00 PM                                                             26/11/23 11:23:52.00 PM

6 rows selected.

SQL> SELECT * FROM ControlEnvio;

COD_CONTROL_ENVIO COD_FARMACIA_CTRL PRECIO_PROMEDIO DISTANCIA_PROMEDIO PESO_PROMEDIO SUMA_PRECIO SUMA_DISTANCIA  SUMA_PESO
----------------- ----------------- -------------- ------------------ ------------- ----------- -------------- ----------
                1                 1
                2                 2              5                .98             7           5            .98          7
                3                 3              5                .99             2           5            .99          2
                4                 4
                5                 5            7.5              2.495            14          15           4.99         28
                6                 6              5                .59            15           5            .59         15
                7                 7              5                .94             8           5            .94          8
```

Como se ve todo se compilo y se llamaron los procedimiento con los bloques anónimos sin errores y manteniendo la integridad de la base de datos.