1. **Crea las siguientes tablas:**

```
CREATE TABLE Contratos(Referencia   VARCHAR(10) PRIMARY KEY,
                       Empresa      VARCHAR(100),
                       Fecha        DATE,
                       NumTrayectos NUMBER(2,0));


CREATE TABLE Trayectos(Referencia VARCHAR(10) REFERENCES Contratos ON
DELETE CASCADE,
                      Origen     VARCHAR(50),
                      Destino    VARCHAR(50),
                      Vehículo   VARCHAR(20),
                      PRIMARY KEY (Referencia, Origen, Destino));
```

a. Escribir un procedimiento que reciba una referencia de contrato (que se asume que existe) como parámetro de entrada y actualice su contador de trayectos (NumTrayectos) con el número que tiene registrados en la tabla Trayectos y lo imprima por consola. Se debe declarar una excepción que se lance para dar un mensaje si la referencia no tiene trayectos.
b. Crear un trigger que mantenga actualizado el atributo redundante NumTrayectos al insertar y borrar en la tabla (se asume que antes de la actualización este atributo es consistente).

2. **Crea las siguientes tablas:**

```
CREATE TABLE Empleados( DNI   CHAR(9) PRIMARY KEY,
                        Nombre      VARCHAR(100),
                        CodDept     CHAR(5) REFERENCES Departamentos on
                  delete set NULL,
                        Salario     NUMBER(4,0));

CREATE TABLE Departamentos(CodDept   CHAR(5) PRIMARY KEY,
                           Nombre      VARCHAR(100));


CREATE TABLE Cambios(IdCambio VARCHAR(10) PRIMARY KEY,
                     Usuario VARCHAR(12),
                      SalarioAnt   NUMBER(4,0),
                       SalarioNew  NUMBER(4,0));
```

a. Implementar un trigger que registre en la tabla Cambios cualquier modificación que se produzca en el salario de un empleado, indicando el usuario en la que se realizó. El identificador se obtendrá de una secuencia denominada SEQCambios.
b. Escribir un procedimiento almacenado que liste por departamento el nombre y salario de cada empleado cuyo salario sea inferior a la media del departamento. Incluir el total de dichos salarios por departamento

3. Crea las siguientes tablas:

```
Create table Autor (
DNI CHAR(9) PRIMARY KEY,
Nombre VARCHAR(50) NOT NULL,
Apellido VARCHAR(50) NOT NULL);
Pais VARCHAR(30) NOT NULL);
NumArticulos NUMBER(3,0) NOT NULL);

Create table Revista (
ISSN VARCHAR(9) PRIMARY KEY,
Nombre VARCHAR(100) NOT NULL);

Create table Articulo (
DOI CHAR(30) PRIMARY KEY,
Titulo VARCHAR(100) NOT NULL,
ISSNRevista VARCHAR(9) NOT NULL REFERENCES Revista(ISSN) ON DELETE
CASCADE,
NumAutores NUMBER(1,0) NOT NULL,
CHECK NumAutores BETWEEN 1 AND 4);

Create table Firma (
DNI CHAR(9) NOT NULL REFERENCES Autor,
DOI CHAR(30) NOT NULL REFERENCES Articulo ON DELETE CASCADE,
PRIMARY KEY(DNI, DOI));
```

a. Escribir un procedimiento almacenado que reciba por argumento el nombre de una revista y muestre por consola los datos de la revista (ISSN, Nombre) y los nombres y apellidos de todos sus autores. Si no tiene autores debe indicar 'No tiene autores'.
b. Implementar un trigger que mantenga actualizada la columas NúmAutores.

4. Crea las siguientes tablas:

```
Create table Aeropuerto(
Codigo CHAR(6) PRIMARY KEY,
Nombre VARCHAR(30) NOT NULL,
Pais VARCHAR(30)NOT NULL);

Create table Vuelo(
Numero CHAR(6),
Fecha DATE,
Origen CHAR(6) NOT NULL REFERENCES Aeropuerto on delete set NULL,
Destino CHAR(6) NOT NULL REFERENCES Aeropuerto on delete set NULL,
Importe NUMBER(6,2),
Plazas NUMBER(3) DEFAULT 100,
primary key (numero,fecha),
unique (fecha, origen, destino),
check(origen<>destino));


Create table Billetes(
Numero CHAR(6),
Fecha DATE NOT NULL,
Pasaporte CHAR(10) NOT NULL,
PRIMARY KEY(Numero, fecha, pasaporte),
FOREIGN KEY(Numero, fecha) REFERENCES vuelo);
```

```
Create table Ventas(
Numero CHAR(6),
Fecha DATE,
Importe NUMBER(6,2),
Vendidos NUMBER(3) DEFAULT 0,
primary key (Numero, Fecha),
foreign key (Numero, Fecha) REFERENCES Vuelo);
```

a. Escribir un procedimiento almacenado que reciba por argumentos una fecha, los códigos de un aeropuerto de origen y uno de destino y un pasaporte y registre un billete en el primer vuelo en el que haya plazas libres. En caso de que no haya vuelos disponibles se informará mediante un mensaje.
b. Implementar un trigger que registre en la tabla Ventas el número total de billetes vendidos y el importe total de las ventas para cada vuelo. En el caso de devolución de un billete tan solo se reintegrará un importe fijo de 150€, no el importe total del billete.

5. Ejecutar las siguientes instrucciones:

```
drop table ComisionCC;
drop table deposito;
drop table log;
create table ComisionCC(cc char(20), importe number(10,2));
create table deposito(cc char(20));
create table log( msg varchar(50));
```

Diseñar un trigger asociado a la operación delete de la tabla ComisionCC, de modo que si la cuenta del registro que se borre se encuentra en la tabla deposito indique en log un mensaje que indique la cc, el importe y el texto “Deposito asociado”. En caso contrario el texto indicará “Cliente preferente”

Hacer las siguientes pruebas para comprobar el funcionamiento:

```
insert into Comisioncc values ('12345678900987654321',13.9);
insert into Comisioncc values('12345123131333344321',13.0);
insert into Comisioncc values ('37423462487654321478',13.9);
insert into deposito values ('37423462487654321478');
delete from ComisionCC;
```

6. **Ejecutar las siguientes instrucciones:**

```
drop table Records;
drop table Marcas;
create table Records(prueba number  primary key,  tiempo number);
create table Marcas(prueba number, fecha date, tiempo number, primary key
(prueba,fecha));
```

Diseñar un trigger asociado a la operación de inserción de la tabla Marcas, de modo que si el tiempo de la prueba que se inserte es un nuevo record se actualice el registro correspondiente en la tabla Records.

Realiza las siguientes pruebas

```
delete from Marcas;
delete from Records;
insert into Marcas values (1, to_date('01/02/2013'),3.8);
insert into Marcas values (1, to_date('02/02/2013'),4.2);
insert into Marcas values (1, to_date('03/02/2013'),3.5);
```

7. Ejecutar las siguientes instrucciones:

```
drop table Libros cascade constraints;
drop table Ejemplares cascade constraints;
create table Libros(isbn char(13) primary key,
                    copias integer);
create table Ejemplares(signatura char(5) primary key,
                        isbn char(13) not null,
                        FOREIGN KEY (isbn)
                        REFERENCES Libros);
```

Escribir un trigger asociado a la inserción de filas en Ejemplares, de forma que si el isbn no aparece en Libros, se cree una fila en Libros con dicho isbn y copias con valor 1, de forma que se evite el error por la violación de la foreign key. En caso de existir, el número de ejemplares se incrementará en uno. Prueba insertando Ejemplares que satisfagan ambas condiciones.