Facultad de Ciencias Exactas y Naturales y Agrimensura

TRABAJO PRACTICO 1, 2, 3 Y 4

BASE DE DATOS 2

INTEGRANTES GRUPO N° 1:

- Cáceres Braun, Juan Gabriel DNI: 41015736
- Kairuz, David Elias DNI: 34973195
- Leiva Falcón, Ayelen DNI: 41281526
- Maldonado Gauna, Esteban Joel DNI: 40350454
- Piriz, Andrés Nahuel DNI: 40586477
- Sandoval, Liz DNI: 41335142

2022

TP 1

Bases de datos 2

1) CREATE TABLE Empleados(
dni int Primary Key,
legajo Char(6) Unique,
ape_y_nom Varchar(60),
cargo Int Check(cargo between 50 and 120),
sueldo Dec(8,2) Check(sueldo <= 90000)
);

- CREATE DOMAIN Documento AS int CHECK (dni > 0);
  ALTER TABLE Empleados ALTER COLUMN dni SET DATA TYPE Documento;

- Create or replace Trigger Control_sueldo
  Before Update of sueldo on Empleados
  FOR EACH ROW
  Begin
  If :new.sueldo > (:old.sueldo * 1.20)
  Then “Error-Debe ir sentencia que cancela la actualización”
  End if
  End;

- Create Trigger Adicional_cargo
  After Insert or Update of cargo on Empleados
  For each row
  Begin
  If :new.cargo = 90
  Then :new.sueldo = :new.sueldo + (Select importe_basico from Importes_basicos where cargo = :new.cargo) * 0.15
  End if
  End;

2) Create table Seguros_autos(
Nro_poliza Char(8) Primary Key,
Dni Char(6) references Clientes(Dni),
Marca_auto Char(7) Check(Marca_auto In ('Ford','Renault','Fiat','Peugeot','VW', 'Toyota', 'Nissan')),
Anio_modelo Char(4) default ('2019'),
Nro_patente Char(10),
Nro_motor Varchar(30),
Periodo_pago Char (6) default ('202001'),
Imp_pagar Dec(10,2) not null,
Foreign Key(marca,anio_modelo) references Vehiculos(marca,anio_modelo)
);

- Create Trigger Facturas_mensuales
  After Insert on Seguros_autos
  For each row
    Declare importe_cuota Dec(10,2)
    Declare nro_cuota integer
    Declare comprobante Char (16)
    Declare fecha_pago_aux Char (8)
    Begin
      importe_cuota= :new.imp_pagar / 3
      nro_cuota = 1
      While nro_cuota < 4
      Loop
      nro_comprobante = :new.Nro_poliza + :new.Periodo_pago
    +convert(char(2), nro_cuota))
      fecha_pago = ´15' + convert(char(2), nro_cuota)) + substring
    (:new.Periodo_pago,3,2)

      Insert into Cuotas_seguros values (nro_comprobante,
    importe_cuota, convert(fecha_pago, date))
      nro_cuota = nro_cuota + 1
      End Loop
    End;

3) Create table Pedidos(
  Nro_orden Char(6) not null unique,
  Fecha_pedido Date default today,
  Cuit_cliente Char(11) not null,
  Cuit_proveedor Char(11) not null,
  Cuit_cliente references Clientes(Cuit_cliente),
  Primary key(Cuit_cliente, nro_orden),
  Nro_producto int,
  Tipo_pedido Char(1) Check(Tipo_pedido In ('M','C','G','H','N')),
  Cant_pedida int Check(Cant_pedida > 0)
  );

- Create Trigger Control_pedido
  Before Insert on Pedidos
  For each row
    Begin
      If :new.cant_pedida > (Select stock_actual from Stock_productos
    where producto = :new.nro_producto)
        Then RollBack
      Else
        Update Stock_productos
          Set stock_actual = stock_actual -
:new.cant_pedida
          where producto= :new.nro_producto
      End if;
    End;

4)

- Alter table Pedidos add Constraint Clave_Producto Foreign Key (nro_producto,tipo_pedido) references Productos(nro_producto,tipo_pedido) Constraint CantidadPermitida check(cant_pedida between 10 and 2000)

  Alter table Pedidos with no check add Constraint ControlCuit check (Cuit_cliente<>Cuit_proveedor)

- Alter table Stock_productos add Constraint ControlStock check (stock_actual < stock_minimo)

5) Create Domain CodigoCliente Int(9) check (CodigoCliente> 0);

```
Create table Clientes(
Cod_cliente CodigoCliente primary key,
Nombre_cli Varchar(40) not null,
Direccion Varchar(35) not null,
Cod_postal Char(5),
Foreign key (cod_postal) references CodPostales(cod_postal),
Importe_base Dec(8,2) not null
);

Create table Facturas(
Nro_factura Char(8) primary key,
Fecha_fact Date,
Cliente_factura CodigoCliente,
Foreign key (cliente_factura) references Clientes(cliente_factura),
Tipo_descuento int(2) check(tipo_descuento between 5 and 20),
Iva int(2) check((iva=15) or (iva=21)),
Importe_factura Dec(10,2) not null
);

Create Trigger Calcular_importe_factura
After Insert or Update importe_base on Clientes
For each row
    Declare resultado Dec(10,2)
    Begin
            resultado = :new.importe_base – ((:new.tipo_descuento *
:new.importe_base)/100)
            resultado = resultado + ((:new.iva * resultado)/100)
            Update Facturas
            Set importe_factura = resultado
            Where cliente_factura = :new.cod_cliente
    End;
```

6) Create table Empleado_baja(

```
Dni Char(8),
Legajo Char(6),
Cargo number,
Usuario Varchar(15),
Fecha Date
```

```
);

Create trigger Empleado_eliminado
After delete on Empleados
For each row
    Begin
        Insert into Empleados_baja values (:old.Dni, :old.Legajo, :old.Cargo,
User, Sysdate)
    End;

Create trigger Baja_usuario
After delete on Empleados
    Begin
        Delete from Usuarios where usuarios.dni= :old.Dni;
    End;

7)  Replace trigger Baja_usuario
After delete on Empleados
    Begin
        Delete from Usuarios where usuarios.legajo= :old.legajo;
    End;

Alter trigger Calcular_importe disable;
Alter table Consumos disable all triggers;

8)  Alter table Facturas Add fecha_vto date;
Alter table Facturas Add fecha_pago date;
Alter table Facturas Add intereses int;

Create Trigger Recargo_factura
After Update fecha_pago on Facturas
For each row
    Begin
        If fecha_pago > fecha_vto then
            Update Facturas
            Set intereses = 5
            Where cliente_factura= :new.cod_cliente
        End if
    End;

9)  Create Trigger Importe_recargo
After Update intereses on Facturas
For each row
    Begin
        Update Facturas
        Set importe_factura = importe_factura + ((importe_factura *
intereses)/100)
        Where cliente_factura= :new.cod_cliente
    End;
```

10) Create table cursadas(
dni int primary key,
cod_carrera int,
cod_materia int,
anio_cursado int not null,
condicion char(1) (check condicion in ('R', 'P', 'D')),
nota_final int (check between 1 and 10),
constraint control_materia foreign key (cod_carrera, cod_materia)
references plan_carreras (cod_carrera, cod_materia)
);

```
Create trigger condicion_alumno
after insert on cursadas or update nota_final on cursadas
for each row
    begin
        if new.nota_final = 6 then
            update cursadas
            set condicion = ‘R’
            where dni = new.dni
        else if new.nota_final >= 7 then
            update cursadas
            set condicion = ‘P’
            where dni = new.dni
        else
            update cursadas
            set condicion = ‘D’
            where dni = new.dni
        End if
    end;
```

Serie Ejercicios Prácticos 2

Bases de Datos Distribuidas

Algoritmos de optimización de consultas

1) Nodo 1: Clientes

Nro. Cliente	Apellido y Nombres	Dirección	Correo electrónico	Fecha de Nacimiento	Nro. de Sucursal
6 bytes	30 bytes	35 bytes	40 bytes	8 bytes	4 bytes

- Contiene: 5000 registros
- Longitud del registro: 123 bytes

Nodo 2: Sucursales

Nro. Sucursal	Nombre de la sucursal	Dirección	Código de Provincia
4bytes	30 bytes	35 bytes	2 bytes

- Contiene: 100 registros
- Longitud del registro: 71 bytes

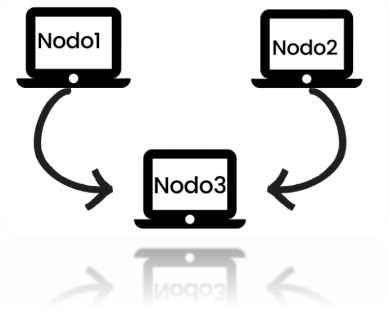
a) El tamaño de la relación Clientes (Nodo 1): 5000 tuplas * 123 B= 615000B.
El tamaño de la relación Sucursales (Nodo 2): 100 tuplas * 71 B= 7100B.

b) Select C.ApellidosNombre, S.NombreSucursal
From Clientes C
Inner join Sucursales S
On C.nroSucursal = S.nroSucursal

El tamaño de la consulta es: (30B + 30B) * 5000tuplas = 300000B

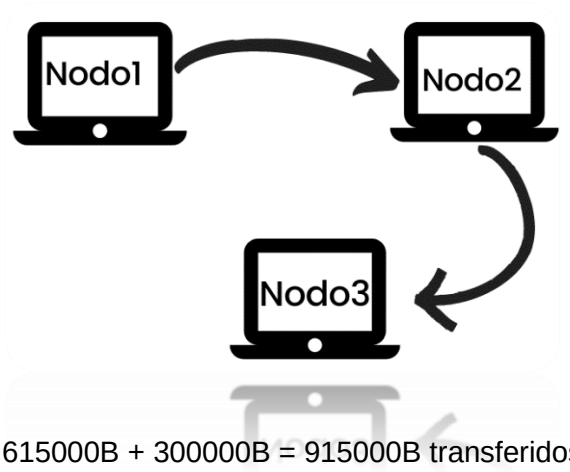
i. Nodo 3; resultado

1ra. Opción: Transferimos el Nodo1 y el Nodo2 al Nodo3:



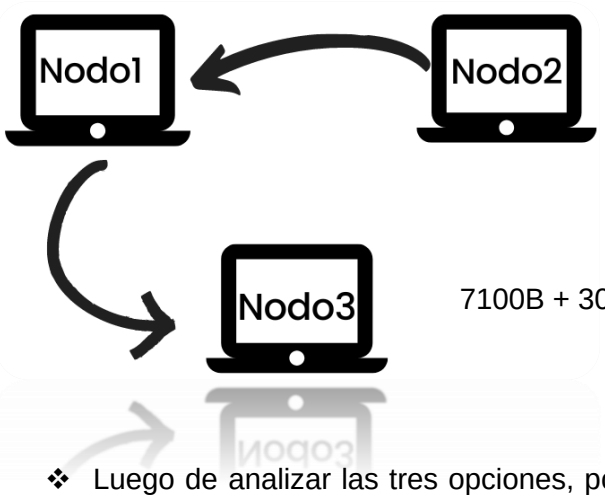
615000B + 7100B = 622100B. transferidos.

2da. Opción: Se transfiere el Nodo1 al Nodo2, donde se ejecuta la sentencia de la consulta y transferimos el resultado al Nodo3 para su visualización:



615000B + 300000B = 915000B transferidos.

3ra.Opción: Se transfiere la información del Nodo2 al Nodo1, donde se ejecuta la sentencia de la consulta y transferimos el resultado al Nodo3 para la visualización:



7100B + 300000B = 307100B transferidos.

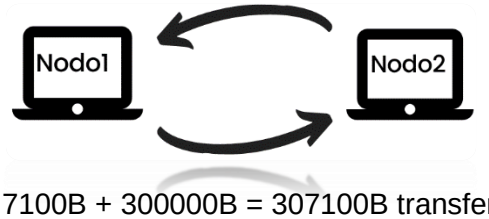
- Luego de analizar las tres opciones, podemos observar que la 3ª opción es la estrategia más adecuada, dado que implica menos transferencia de datos.

ii. Nodo2, como nodo resultado:

1ra. Opción: Se transfiere el Nodo1 al Nodo2, donde se ejecuta la sentencia de la consulta y se visualizará el resultado:

En esta opción el tamaño de bytes transferidos es 615000B, es decir, el tamaño de la relación Clientes.

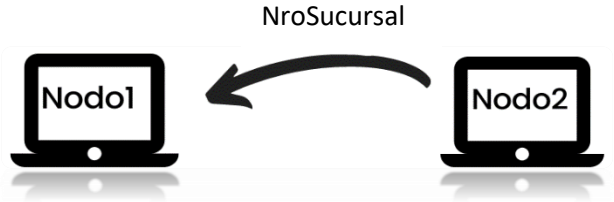
2da. Opción: Se transfiere el Nodo2 al Nodo1, y de este mismo se transfiere el resultado de la consulta al Nodo2 en el cual se visualizará el resultado:



7100B + 300000B = 307100B transferidos.

- ❖ Siendo esta última opción la estrategia mas optima, dado que implica menor trasferencia de los datos.

iii. …semi reunión (semi join):



4B * 100tuplas = 400B transferidos.

(4B + 30B) * 5000tuplas = 170000B transferidos.

- ❖ Como podemos observar en semi join transferimos solamente aquellos atributos necesarios para ejecutar nuestra consulta, para obtener el total de datos transferidos debemos sumar ambos resultados:
  400B + 170000B = 170400B transferidos. Resultando la más optima que lo analizado en el punto anterior (ii.).

2)

Nodo 1 : Profesores

DNI	Apellido y Nombres	Correo electrónico	Fecha de Nacimiento	Cód. de Facultad
8 bytes	30 bytes	40 bytes	8 bytes	2 bytes

Contiene: 7000 registros
Longitud del registro: 88 bytes

Nodo 2 : Facultades

Cód. Facultad	Denominación	Dirección	Decano
2 bytes	32 bytes	35 bytes	8 bytes

Contiene: 200 registros
Longitud del registro: 77 bytes

a) _El tamaño de la relación Profesores (Nodo 1): 7000 tuplas * 88B= 616000B.

El tamaño de la relación Facultades (Nodo 2): 200 tuplas * 77B= 15400B.

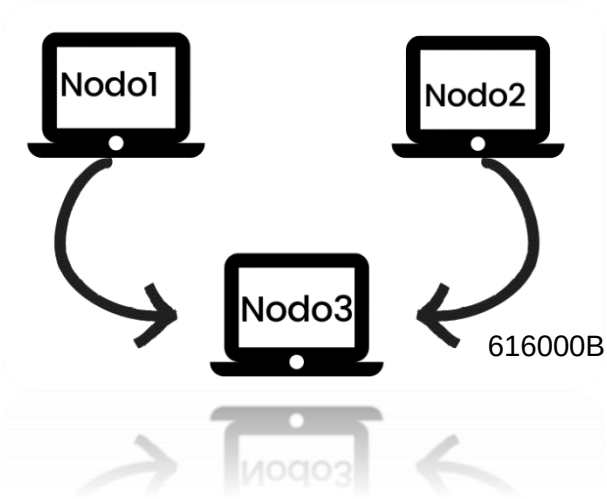
b) _Select P.ApellidosNombres, F.Denominacion

From Profesores P
Inner join Facultades F
On P.codFacultad = F.codFacultad

El tamaño de la consulta es: (30B + 32B) * 7000tuplas = 434000B.

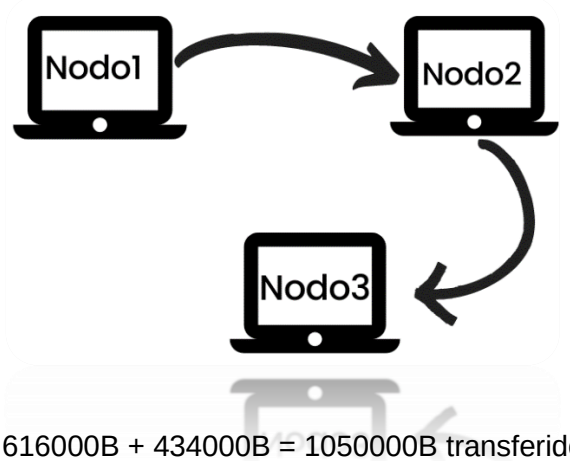
i. Nodo 3, como nodo resultado:

1ra. Opción: Transferir tanto el Nodo1 como el Nodo2 al nodo resultado (Nodo3):



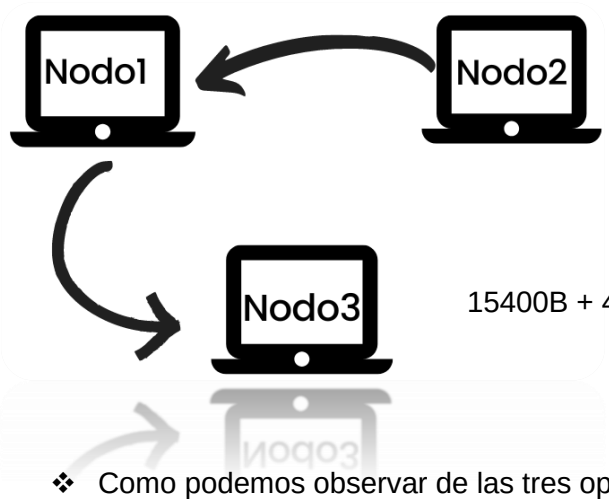
616000B + 15400B = 631400B. transferidos.

2da. Opción: Se transferir el Nodo1 al Nodo2, donde se ejecuta la sentencia de la consulta y transferimos el resultado al Nodo3 para su visualización:



616000B + 434000B = 1050000B transferidos.

3ra.Opción: Transferir el Nodo2 al Nodo1, donde se ejecuta la sentencia de la consulta, transfiriendo el resultado al Nodo3 para la visualización:

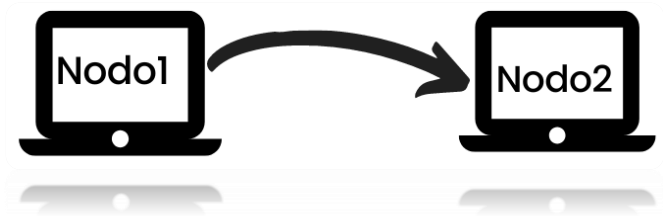


15400B + 434000B = 449400B transferidos.

- Como podemos observar de las tres opciones, la 3ª opción es la estrategia más optima de acuerdo al criterio considerado, siendo esta la que implica menor transferencia de datos.

ii_Nodo2, como nodo resultado:

1ra. Opción: Se transfiere el Nodo1 al Nodo2, donde se ejecuta la sentencia de la consulta y se visualizará el resultado:



En esta opción el tamaño de bytes transferidos es 616000B, es decir, el tamaño de la relación Profesores.

2da. Opción: Se transfiere el Nodo2 al Nodo1, el cual transfiere el resultado de la consulta al Nodo2 para su visualización:

15400B + 434000B = 449400B transferidos.

- Como podemos observar esta opción resulta la más optima, dado que implica menor transferencia de los datos.

iii_…semi reunión (semi join):

CodFacultad

2B * 200tuplas = 400B transferidos.

ApellidoyNombres, CodFacultad

(30B + 2B) * 7000tuplas = 224000B transferidos.

- Como podemos observar en semi join transferimos solo aquellos atributos necesarios para ejecutar nuestra consulta, obteniendo el total de datos transferidos sumando ambos resultados: 400B + 224000B = 224400B transferidos. Resultando la más optima incluso que lo obtenido en el punto anterior (2_b_ii.).

3) **Nodo 1 : Remedios**

Producto	Nombre Comercial	Precio	Acción terapéutica	Laboratorio
6 bytes	20 bytes	10 bytes	40 bytes	5 bytes

Contiene: 100000 registros Longitud del registro: 81 bytes

Nodo 2 : Laboratorios

Laboratorio	Nombre	Dirección	Sitio web	Gerente	Teléfono
5 bytes	40 bytes	35 bytes	40 bytes	45 bytes	12 bytes

Contiene: 63800 registros Longitud del registro: 177 bytes

a) _El tamaño de la relación Remedios (Nodo 1): 100000 tuplas * 81B= 8100000B.

El tamaño de la relación Laboratorios (Nodo 2): 63800 tuplas * 177B= 11292600B.

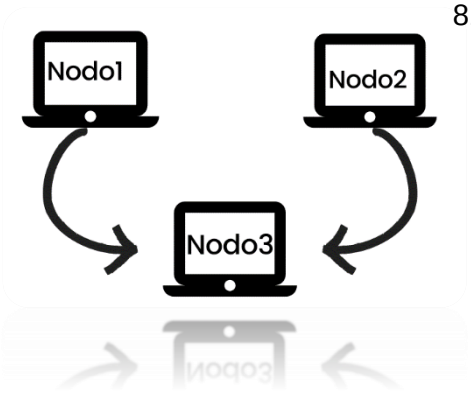
b) _Select R.nombreComercial, L.Nombre

From Remedios R
Inner join Laboratorios L
On R.Laboratorio = L.Laboratorio

El tamaño de la consulta es: (20B + 40B) * 100000tuplas = 6000000B.

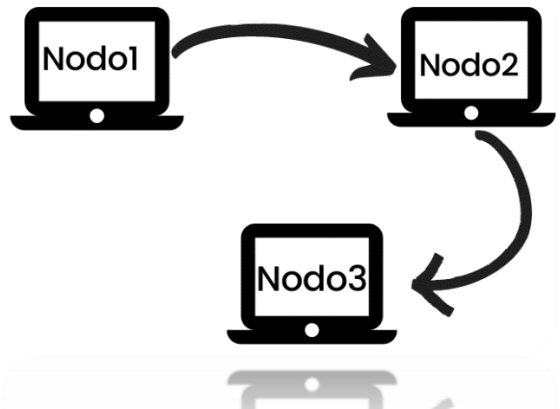
ii. Nodo 3, como nodo resultado:

1ra. Opción: Transferir tanto el Nodo1 como el Nodo2 al nodo resultado (Nodo3):



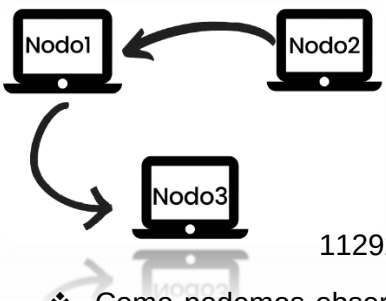
8100000B + 11292600B = 19392600B. transferidos.

2da. Opción: Transferir el Nodo1 al Nodo2, donde se ejecuta la consulta y se transfiere el resultado al Nodo3 para la visualización:



8100000B + 6000000B = 14100000B transferidos.

3ra.Opción: Transferir el Nodo2 al Nodo1, donde se ejecuta la consulta, transfiriendo el resultado al Nodo3 para la visualización:



11292600B + 6000000B = 17292600B transferidos.

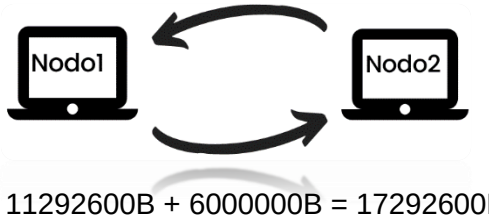
- Como podemos observar de las tres opciones, la 2ª opción es la más optima, teniendo en cuenta el criterio de “minimizar el transporte de bytes, entre los distintos nodos distribuidos”.

ii_Nodo2, como nodo resultado:

1ra. Opción: Se transfiere el Nodo1 al Nodo2, donde se ejecutará la consulta y se visualizará el resultado:

El tamaño de bytes transferidos es igual al tamaño de la relación Remedios: 8100000B.

2da. Opción: Se transfiere el Nodo2 al Nodo1, el cual transfiere el resultado de la consulta al Nodo2 para su visualización:



11292600B + 6000000B = 17292600B transferidos.

- De las dos opciones, la 1ª opción resulta la mas optima, ya que implica menor transferencia de datos.

4)

Nodo 1 : Empleados

DNI	Nombre y Apellido	Fecha Ingreso	Código Departamento	Categoría	Sueldo
8 bytes	25 bytes	8 bytes	3 bytes	2 bytes	10 bytes

Contiene: 3200 registros Longitud del registro: 56 bytes

Nodo 2 : Departamentos

Código	Nombre	Dni Jefe	Sector	Área	Mail
3 bytes	40 bytes	8 bytes	4 bytes	35 bytes	40 bytes

Contiene: 1500 registros Longitud del registro: 130 bytes

a) _El tamaño de la relación Empleados (Nodo 1): 3200 tuplas * 56B= 179200B.

El tamaño de la relación Departamentos (Nodo 2): 1500 tuplas * 130B= 195000B.

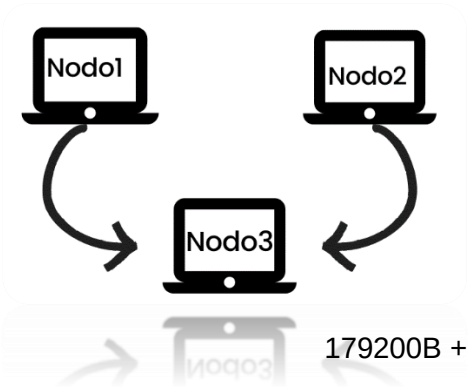
b) _Select NombreApellido, Nombre

From Empleados
Inner join Departamentos
On codDepartamento = codigo

El tamaño de la consulta es: (25B + 40B) * 3200tuplas = 208000B.

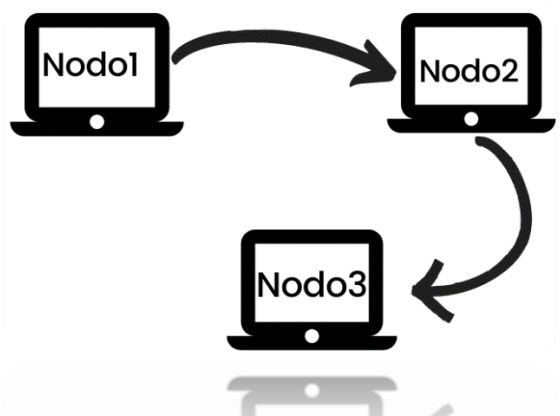
ii_Nodo 3, como nodo resultado:

1ra. Opción: Transferir tanto el Nodo1 como el Nodo2 al nodo resultado (Nodo3):



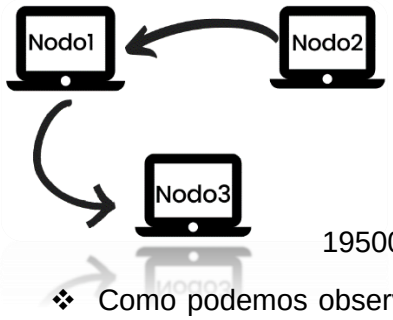
179200B + 195000B = 374200B. transferidos.

2da. Opción: Transferir el Nodo1 al Nodo2, donde se ejecuta la consulta y se transfiere el resultado al Nodo3 para la visualización:



179200B + 208000B = 387200B transferidos.

3ra.Opción: Transferir el Nodo2 al Nodo1, donde se ejecuta la consulta, transfiriendo el resultado al Nodo3 para la visualización:



195000B + 208000B = 403000B transferidos.

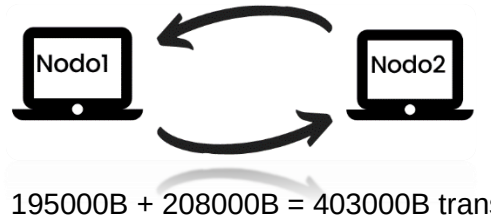
- Como podemos observar de las tres opciones, la 1ª opción es la más optima, teniendo en cuenta el criterio de "minimizar el transporte de bytes, entre los distintos nodos distribuidos".

ii_Nodo2, como nodo resultado:

1ra. Opción: Se transfiere el Nodo1 al Nodo2, donde se ejecutará la consulta y se visualizará el resultado:

El tamaño de bytes transferidos es igual al tamaño de la relación Empleados: 179200B.

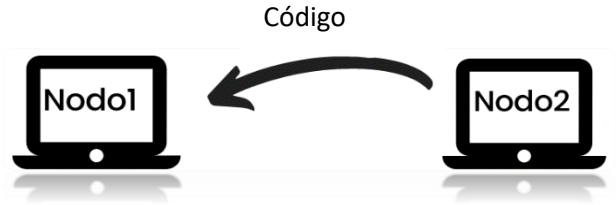
2da. Opción: Se transfiere el Nodo2 al Nodo1, el cual transfiere el resultado de la consulta al Nodo2 para su visualización:



195000B + 208000B = 403000B transferidos.

- De las dos opciones, la 1ª opción resulta la más optima, ya que implica menor transferencia de datos.

iii_…semi reunión (semi join):



3B * 1500tuplas = 4500B transferidos.

(25B + 3B) * 3200tuplas = 89600B transferidos.

- Como podemos observar en semi join transferimos solo aquellos atributos necesarios para ejecutar la consulta, obteniendo el total de datos transferidos sumando ambos resultados: 4500B + 89600B = 94100B transferidos. Resultando la más optima incluso que lo obtenido en el punto anterior (4_b_ii.).

5)

Nodo 1 : Vehículos

Nro. Patente	Nombre y Apellido del Titular	DNI	Dirección	Código de Modelo	Registro Seccional
8 bytes	25 bytes	8 bytes	25 bytes	5 bytes	4 bytes

Contiene: 1300000 registros Longitud del registro: 73 bytes

Nodo 2 : Modelos (de los vehículos)

Código	Nombre Modelo	Año	País de origen	Precio
5 bytes	20 bytes	4 bytes	20 bytes	13 bytes

Contiene: 15300 registros Longitud del registro: 62

a) _El tamaño de la relación Vehículos (Nodo 1): 1300000 tuplas * 73B= 94900000B.

El tamaño de la relación Modelos (Nodo 2): 15300 tuplas * 62B= 948600B.

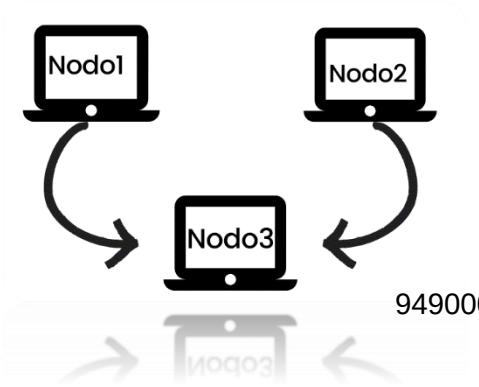
b) _Select V.NomApeTitular, M.NombreModelo, M.Año

```
From Vehículos V
Inner join Modelos M
On  V.codModelo = M.codigo
```

El tamaño de la consulta es: (25B + 20B + 4B) * 1300000tuplas = 63700000B.

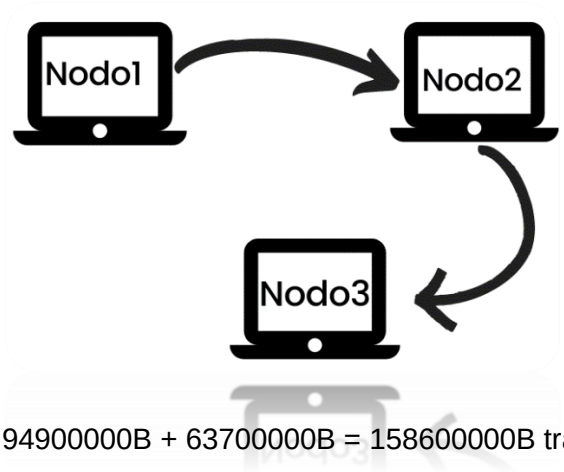
ii_Nodo 3, como nodo resultado:

1ra. Opción: Transferir tanto el Nodo1 como el Nodo2 al nodo resultado (Nodo3):



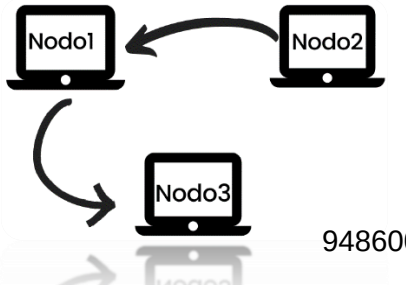
94900000B + 948600B = 95848600B. transferidos.

2da. Opción: Transferir el Nodo1 al Nodo2, donde se ejecutará la consulta, transfiriendo el resultado al Nodo3 para la visualización:



94900000B + 63700000B = 158600000B transferidos.

3ra.Opción: Transferir el Nodo2 al Nodo1, donde se ejecuta la consulta y se transfiere el resultado al Nodo3 para la visualización:



948600B + 63700000B = 64648600B transferidos.

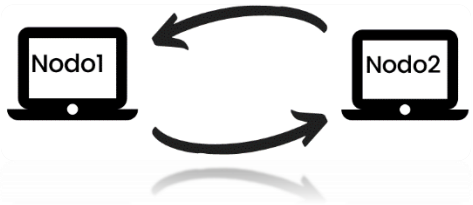
- Como podemos observar de las tres opciones, la 3ª opción es la más optima, teniendo en cuenta el criterio de “minimizar el transporte de bytes, entre los distintos nodos distribuidos”.

ii_Nodo2, como nodo resultado:

1ra. Opción: Se transfiere el Nodo1 al Nodo2, donde se ejecutará la consulta y se visualizará el resultado:

El tamaño de bytes transferidos es igual al tamaño de la relación Vehículos: 94900000B B.

2da. Opción: Se transfiere el Nodo2 al Nodo1, el cual ejecutará la consulta y transferirá el resultado al Nodo2 para la visualización:



948600B + 63700000B = 64648600B transferidos.

- De las dos opciones, la 2ª opción resulta la más optima, ya que implica menor transferencia de datos.

Serie Ejercicios Prácticos 3
Bases de Datos Orientado a Objetos

Objetos Complejos

Considerar los siguientes datos, para definir los objetos mediante la forma (i,c,v), donde i=identificador del objeto, c=constructor, v=estado o valor actual, contemplando los siguientes constructores: atom, set y tuple.

1) Sean los datos de una empresa de telefonía celular:

Valores atómicos

1) Nro. Empresa: 100
2) Nombre Empresa: Telecom
3) Sucursal1: Posadas
4) Sucursal2: Salta
5) Sucursal3: Formosa
6) Fecha creación: 01-02-1994
7) Dni: 24987422
8) Sueldo: 75000

Conjuntos

9) Sucursales={Sucursal1, Sucursal2, Sucursal3}

Tuplas

10) Empresa (objeto complejo)

Nro. Empresa	Nombre Empresa	Sucursales**(9)**	Fecha Creación	Presidente**(11)**
		tipo set		tipo tuple

11) Presidente

Dni	Sueldo

a) Definir los objetos, teniendo en cuenta los valores y tipos dados
b) Representar gráficamente el objeto complejo Empresa

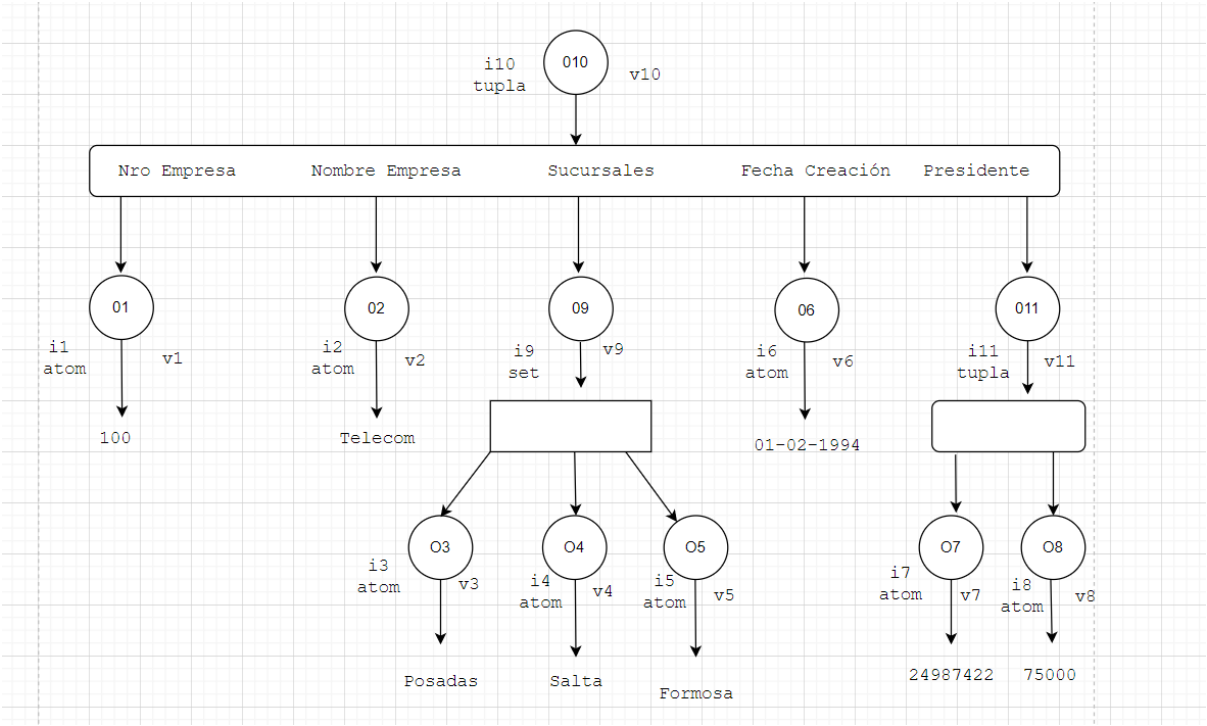
1-a)

O_1 =(I1,atom,100)
O_2 =(I2,atom,Telecom)
O_3 =(I3,atom,Posadas)
O_4 =(I4,atom,Salta)
O_5 =(I5,atom,Formosa)
O_6 =(I6,atom,01-02-1994)
O_7 =(I7,atom,24987422)
O_8 =(I8,atom,35000)

O_9 =(I9,set,{I3,I4,I5})

O_{10} =(I10,tupla,<Nro Empresa, Nombre Empresa, Sucursales(9), Fecha Creacion, Presidente>)
O_{11} =(I11,tupla,<Dni,Sueldo>)

1-b) Representación Gráfica



2) Sean los datos de una facultad:

Valores atómicos

1) Código Facultad: 11
2) Nombre Facultad: Medicina
3) Sede1: Centro
4) Sede2: Campus Cabral
5) Posgrado1: Higiene
6) Posgrado2: Enfermería
7) Posgrado3: Salud Mental
8) Dirección: Moreno 1240
9) Legajo: M1378
10) Nombre: Carlos
11) Apellido: Monti
12) CUIL: 20-18980067-4
13) Mail: cm@med.unne.edu.ar

Conjuntos

14) Sedes = {Sede1, Sede2}
15) Posgrados = {Posgrado1, Posgrado2, Posgrado3}

Tuplas

16) Facultad (objeto complejo)

Código Facultad	Nombre Facultad	Sedes **(14)**	Posgrados **(15)**	Dirección	Decano **(17)**
		tipo set	tipo set		tipo tuple

17) Decanos

Id **(18)**	Legajo	CUIL
tipo tuple		

18) Empleados

Nombre	Apellido	Mail	Lugar trabajo **(16)**
			tipo tuple

a) Definir los objetos, teniendo en cuenta los valores y tipos dados
b) Representar gráficamente el objeto complejo Facultad

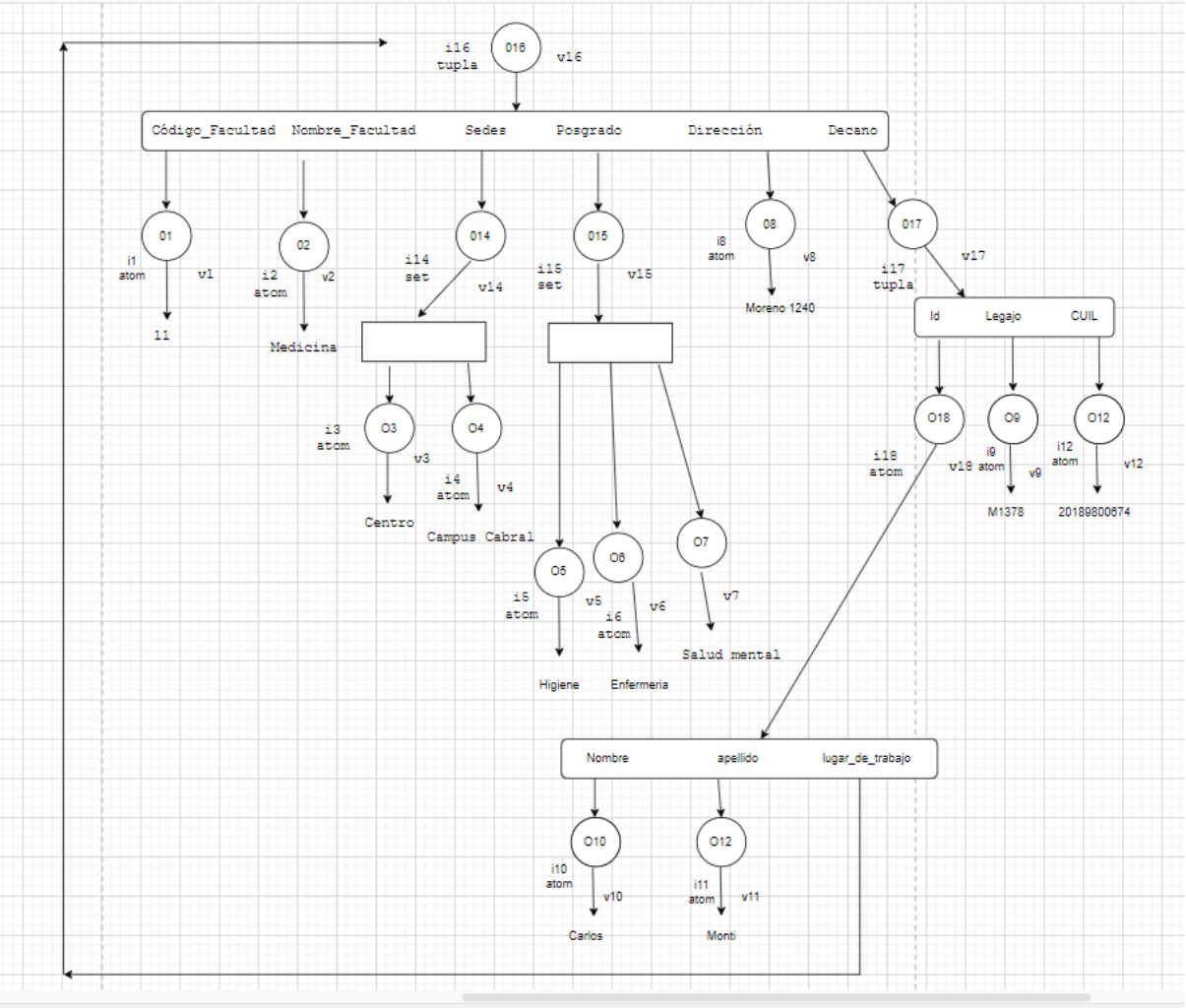
2-a)

Valores atómicos
O_1 =(I1,atom,11)
O_2 =(I2,atom, Medicina)
O_3 =(I3,atom, Centro)
O_4 =(I4,atom,Campus Cabral)
O_5 =(I5,atom, Higiene)
O_6 =(I6,atom, Enfermería)
O_7 =(I7,atom,Salud mental)
O_8 =(I8,atom,Moreno 1240)
O_9 =(I9,atom, M1378)
O_{10} =(I10,atom, Carlos)
O_{11} =(I11,atom,Monti)
O_{12}=(I12,atom,20-18980067-4)
O_{13} =(I13,atom,'cm@med.unne.edu.ar')

Conjuntos
Sedes
O_{14} =(I14, set , {I3,I4})
Posgrado
O_{15} =(I15, set , {I5,I6,I7})

Tuplas
Facultad
O_{16} =(I16, tupla,<Codigo Facultad: I1, Nombre Facultad: I2, Sedes : I14, Posgrado : I15, Dirección: I8, Decano : I17 >)
Decano
O_{17} =(I17, tupla ,<ID: I18, Legajo : I9,CUIL : I12>)
Empleados
O_{18} =(I18, tupla ,< Nombre: I10, Apellido : I11, Lugar Trabajo :I16 >)

2-b) Representación Gráfica


i16
tupla
016
v16
Código_Facultad
Nombre_Facultad
Sedes
Posgrado
Dirección
Decano
01
i1
atom
v1
11
02
i2
atom
v2
Medicina
014
i14
set
v14
015
i15
set
v15
08
i8
atom
v8
Moreno 1240
017
i17
tupla
v17
Id
Legajo
CUIL
O18
i18
atom
v18
O9
i9
atom
v9
M1378
O12
i12
atom
v12
20189800674
O3
i3
atom
v3
Centro
O4
i4
atom
v4
Campus Cabral
O5
i5
atom
v5
O6
i6
atom
v6
O7
v7
Salud mental
Higiene
Enfermeria
Nombre
apellido
lugar_de_trabajo
O10
i10
atom
v10
Carlos
O12
i11
atom
v11
Monti

3) Sean los datos de una concesionaria de motos:

Valores atómicos

1) CUIT: 33-19332111-5 2) Nombre Concesionaria: Ghiggeri Motos
3) Marca1: Honda 4) Marca2: Suzuki 5) Marca3: Motomel 6) Marca4: Ghiggeri
7) Marca5: Yamaha 8) Marca6: Guerrero 9) Marca7: Zanella
10) Sitio web: motos.gmmotos.com.ar 11) Ciudad1: Resistencia 12) Ciudad2: Castelli
13) Ciudad3: Barranqueras 14) Ciudad4: Corrientes 15) Ciudad5: Formosa
16) DNI: 25334812 17) Teléfono: 3624441515
18) Mail: julioayala@gmmmotos.com.ar 19) Nombre: Julio 20) Apellido: Ayala
21) Fecha Nacimiento: 25-10-1967

Conjuntos

22) Marcas = {Marca1, Marca2, Marca3, Marca4, Marca5, Marca6, Marca7}
23) Ciudades = {Ciudad1, Ciudad2, Ciudad3, Ciudad4, Ciudad5}

Tuplas

24) Concesionaria (objeto complejo)

CUIT	Nombre Concesionaria	Representantes **(25)**	Marcas **(22)**	Sitio web	Ciudades **(23)**
		tipo tuple	tipo set		tipo set

25) Representantes

Código **(26)**	Dni	Teléfono	Mail

tipo tuple

26) Personal

Nombre	Apellido	Fecha Nacimiento	Concesionaria **(24)**
			tipo tuple

a) Definir los objetos, teniendo en cuenta los valores y tipos dados
b) Representar gráficamente el objeto complejo Concesionaria

3-a)

Valores atómicos

O_1 =(I1,atom,33-19332111-5)
O_2 =(I2,atom, Ghiggeri Motos)
O_3 =(I3,atom, Honda)
O_4 =(I4,atom, Suzuki)
O_5 =(I5,atom, Motomel)
O_6 =(I6,atom, Ghiggeri)
O_7 =(I7,atom,Yamaha)
O_8 =(I8,atom,Guerrero)
O_9 =(I9,atom, Zanella)
O_{10} =(I10,atom, motos.gmmotos.com.ar)
O_{11} =(I11,atom,Resistencia)
O_{12}=(I12,atom,Castelli)
O_{13} =(I13,atom,Barranqueras)
O_{14} =(I14,atom,Corrientes)
O_{15} =(I15,atom,Formosa)
O_{16} =(I16,atom,25334812)
O_{17} =(I17,atom,3624441515)
O_{18} =(I18,atom,Julioayala@gmmotos.com.ar)
O_{19} =(I19,atom,Julio)
O_{20} =(I19,atom,Ayala)

O_{21} =(I21, atom , 25-10-1967)

Conjuntos

Marcas

O_{22} =(I22, set , { I3, I4, I5, I6, I7, I8, I9})

Ciudades

O_{23} =(I23, set , { I11, I12, I13, I14, I18})

Tuplas

Concesionaria

O_{24} =(I24, tupla,<Cuit : I1, Nombre Concesionaria : I2, Representantes : I25, Marcas I22, Sitio Web : I10, Ciudades :I23 >)

Representantes

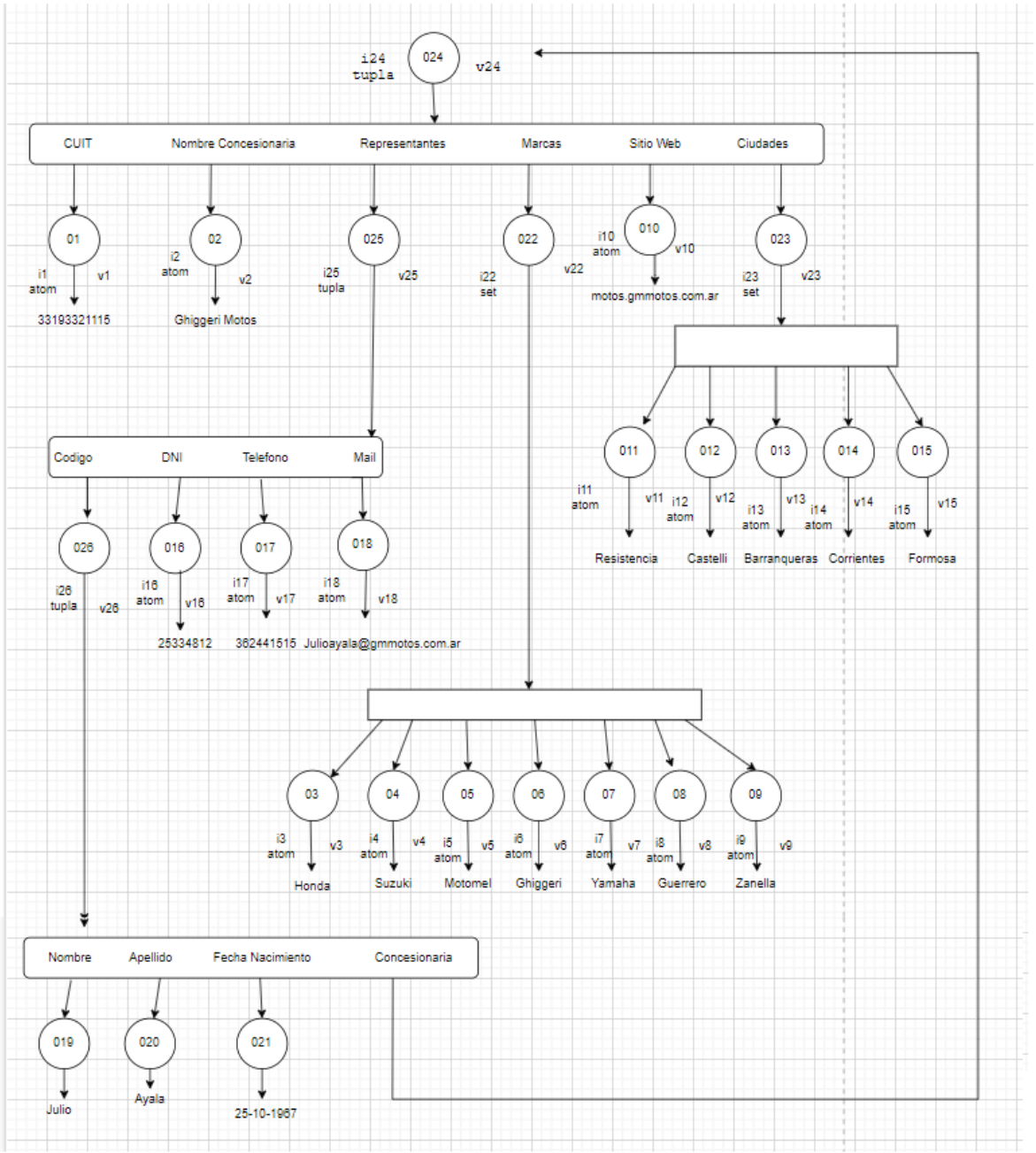
O_{25} =(I25, tupla ,<Código: I26, Dni : I16, Teléfono : I17, Mail :I18>)

Personal

O_{26} =(I26, tupla ,< Nombre: I19, Apellido : I20, Fecha Nacimiento : I21, Concesionaria : I24 >)

3-b)

Representación Gráfica



4) Sean los datos del organismo, Dirección de Rentas de la provincia de Corrientes:

Valores atómicos

1) Nombre organismo: DGR_Ctes 2) Sitio web: www.dgrcorrientes.gob.ar
3) Dependencia1: Saladas 4) Dependencia2: Mercedes 5) Dependencia3: Esquina
6) Dependencia4: Alvear 7) Dni: 21324105 8) Profesión: Contador Público
9) Legajo: R-12542 10) Antigüedad: 25

Conjuntos

11) Dependencias={Dependencia1, Dependencia2, Dependencia3, Dependencia4}

Tuplas

12) Organismo (objeto complejo)

Nombre organismo	Sitio web	Dependencias **(11)**	Director **(13)**
		tipo set	tipo tuple

13) Director

Dni	Profesión	Legajo	Antigüedad

c) Definir los objetos, teniendo en cuenta los valores y tipos dados
d) Representar gráficamente el objeto complejo Organismo

4-a)

Valores atómicos

O_1 =(I1,atom,DGR Ctes)
O_2 =(I2,atom,www.dgrcorrientes.gob.ar)
O_3 =(I3,atom,Saladas)
O_4 =(I4,atom,Mercedes)
O_5 =(I5,atom,Esquina)
O_6 =(I6,atom,Alvear)
O_7 =(I7,atom,21324105)
O_8 =(I8,atom,Contador Público)
O_9 =(I9,atom, R-12542)
O_{10} =(I10,atom, 25)

Conjuntos

Dependencias
O_{11} =(I11, set , { I3, I4 , I5, I6 })

Tuplas

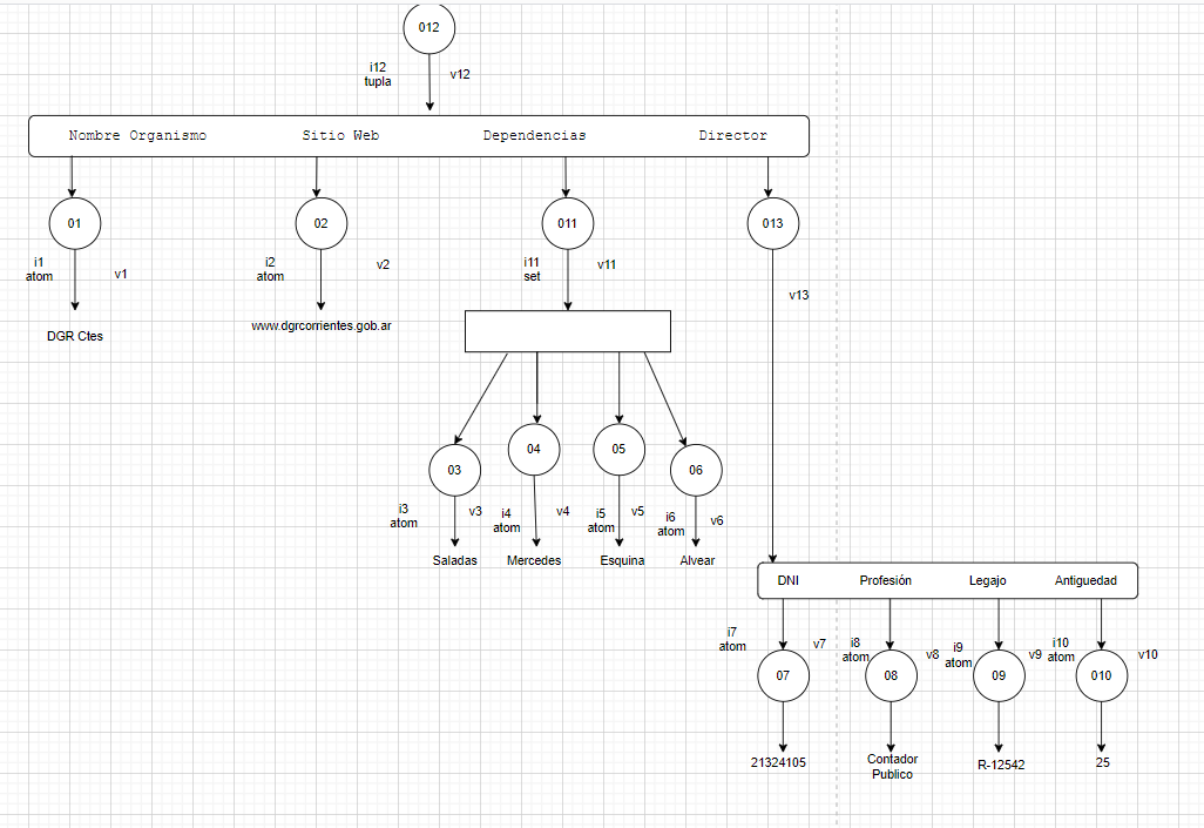
Organismo

O_{12} =(I10,tupla,< Nombre Organismo : I1, Sitio Web : I2 , Dependencias : I11, Director :I13 >)

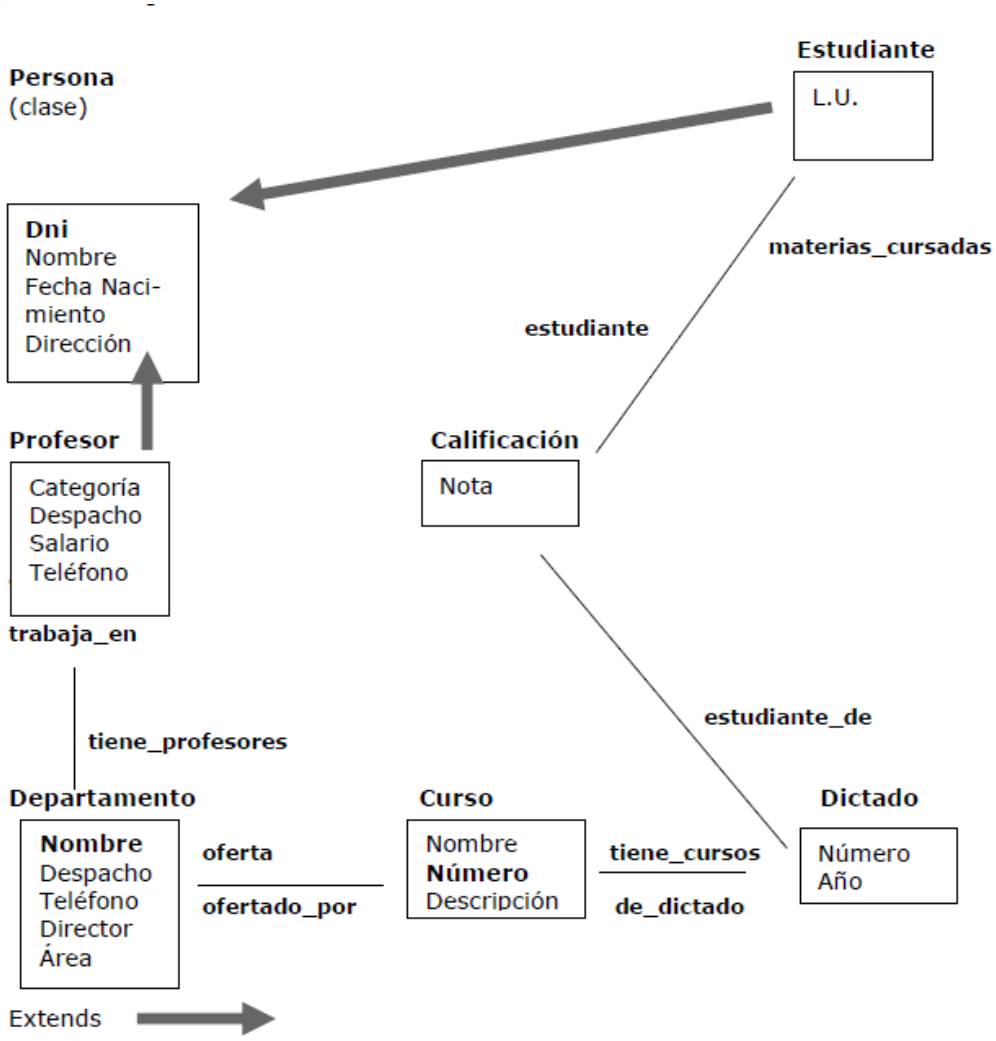
Director

O_{13} =(I11,tupla,<Dni : I2 , Profesión: I8, Legajo :I9, Antigüedad : I10>)

4-b)



5)Definición de clases necesarias para las relaciones :



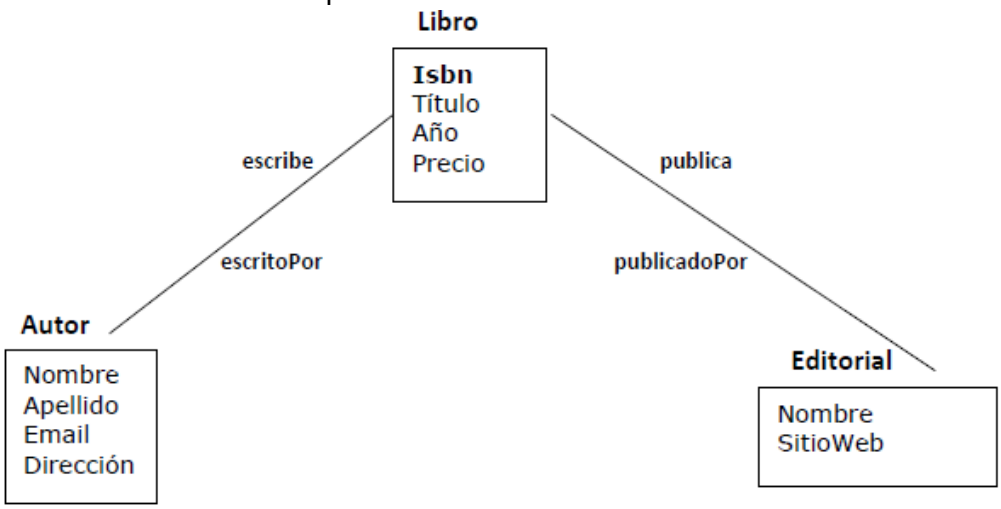
class **Persona** (extent personas key DNI)
{
/* Definición de atributos */
attribute string DNI;
attribute struct nombrePersona {string nombre1, string nombre2, string apellido1, string apellido2} nombre;
attribute date fechanac;
attribute struct direPersona {string calle, integer numero, string codpostal} direccion;
}

class **Profesor** extent Persona (extent profesores)

```
{
/* Def de atributos */
attribute integer categoria;
attribute string despacho;
attribute float salario;
attribute string telefono;
/* Def relaciones */
relationship Departamento trabaja_en
inverse Departamento::tiene_profesores;
}

class Estudiante extent Persona (extent estudiantes)
{
attribute integer LU;
relationship set<Calificacion> materias_cursadas
inverse Calificacion::estudiante;
}

class Departamento (extent departamentos key nombre)
{
attribute string nombre;
attribute string despacho;
attribute string telefono;
attribute Profesor director;
attribute string area;
relationship set<Profesor> tiene_profesores
inverse Profesor::trabaja_en;
relationship set<Curso> oferta
inverse Curso::ofertado_por;
}

class Curso (extent cursos key numero)
{
attribute string nombre;
attribute integer numero;
attribute string descripcion;
relationship set<Dictado> de_dictado
inverse Dictado::tiene_cursos;
relationship Departamento ofertado_por
inverse Departamento::oferta;
}

class Dictado (extent dictados)
```

{
attribute short numero;
attribute integer año;
relationship set<Calificacion> estudiante_de
inverse Calificacion::cursa;
relationship set<Curso> tiene_cursos
inverse Curso::de_dictado;
}
class **Calificación** (extent calificaciones)
{attribute float nota;
relationship Dictado cursa
inverse Dictado::estudiante_de;
relationship Estudiante estudiante
inverse Estudiante::materias_cursadas;
}

6) Definición de clases necesarias para las relaciones :

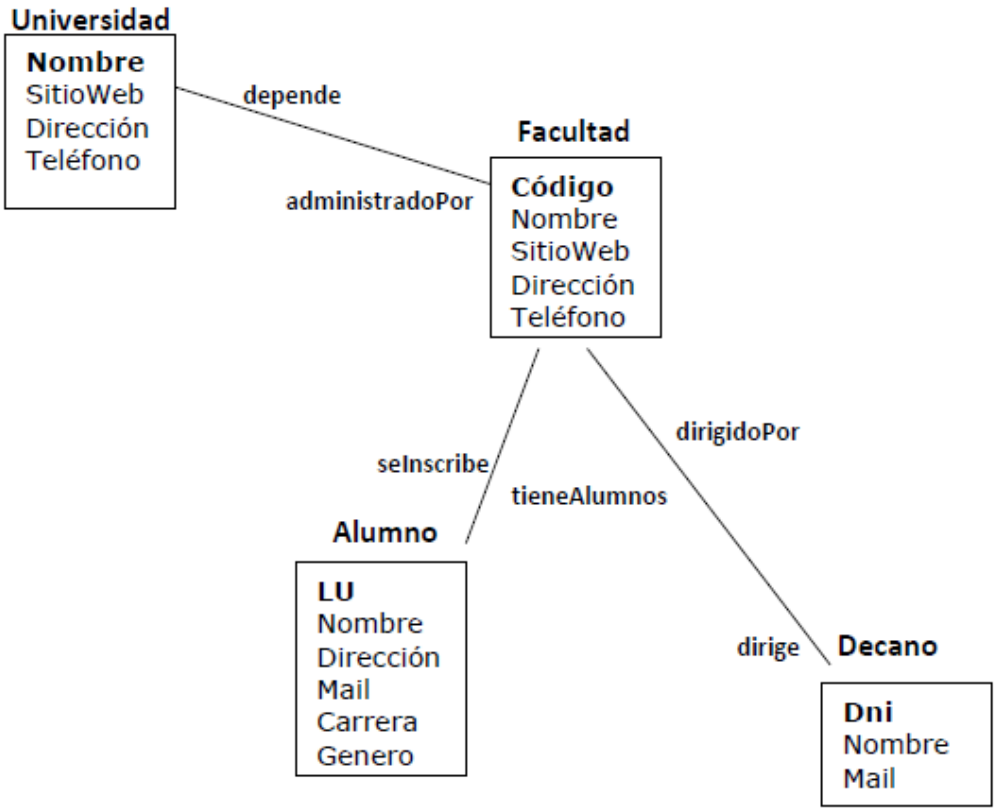


class **Libro** (key ISBN)
{attribute string ISBN;
attribute string titulo;
attribute integer anio;
attribute float precio;
relationship Editorial publicadoPor inverse Editorial::publica;
relationship Autor escritoPorinverse Autor::escribe;
};

class **Editorial** (key nombre)
{attribute string nombre;
attribute string sitioWeb;
relationship Libro publica
inverse Libro::publicadoPor;
};

class **Autor** (key nombreApellido)
{attribute Struct nombreApellido
(string nombre, string apellido);
attribute string email;
attribute Struct direccion
(string calle, integer numero, integer codPostal);
relationship Libro escribe
inverse Libro::escritoPor;
);

7) Definición de clases necesarias para las relaciones :



class **Facultad** (key nombre)
{attribute string nombre;
attribute string sitioWeb;
attribute string direccion;
attribute string telefono;
relationship Universidad administradoPor
inverse Universidad::depende;
relationship Alumno tieneAlumnos
inverse Alumno::seInscribe;
relationship Decano dirigidoPor
inverse Decano::dirige;
};

class **Universidad** (key nombre)
{attribute string nombre;
attribute string sitioWeb;
attribute string direccion;
attribute string telefono;
relationship Facultad depende
inverse Facultad::administradoPor;
};

class **Alumno** (key LU)
{attribute integer LU;
attribute string nombre;
attribute string direccion;
attribute string telefono;
attribute string mail;
attribute string carrera;
attribute Enum genero ('M', 'F');
relationship Facultad selnscribe
inverse Facultad::tieneAlumnos;
};

class **Decano** (key dni)
{attribute integer dni;
attribute integer cuil;
attribute string nombre;
relationship Facultad dirige
inverse Facultad::dirigidoPor;
};

8) Ejercicio complementario Objetos Complejos

Valores atómicos

1) Habilitación club:120571
2) Nombre club: Hindú Club
3) Sede1: Resistencia
4) Sede2: Sáenz Peña
5) Sede3: Charata
6) Deportes1: Básquet
7) Deportes2: Fútbol
8) Deportes3: Natación
9) Dirección: Alvear 1735
10) Legajo: H-1028
11) Nombre: Liliana Beatriz
12) Apellido: Hidalgo
13) CUIL: 23-35812147-4
14) Mail: lbhidalgo@clubhindu.com.ar

Conjuntos

15) Sedes = {Sede1, Sede2, Sede3}
16) Deportes = {Deportes1, Deportes2, Deportes3}

Tuplas

17) **Club deportivo** (objeto complejo)

Habilitación club	Nombre Club	Deportes **(16)**	Sedes **(15)**	Dirección	Presidente **(18)**

18) Presidente

Nombre	Apellido	Mail	Legajo	CUIL

a) Definir los objetos, teniendo en cuenta los valores y tipos dados
b) Representar gráficamente el objeto complejo Club deportivo

8-a)

Valores atómicos

O_1 =(I1,atom,120571)
O_2 =(I2,atom, Hindú Club)
O_3 =(I3,atom,Resistencia)
O_4 =(I4,atom,Sáenz Peña)
O_5 =(I5,atom, Charata)
O_6 =(I6,atom,Básquet)
O_7 =(I7,atom,Fútbol)
O_8 =(I8,atom,Natación)
O_9 =(I9,atom,Alvear 1735)
O_{10} =(I10,atom,H-1028)
O_{11} =(I11,atom, Liliana Beatriz)
O_{12} =(I12,atom,Hidalgo)
O_{13} =(I13,atom, 23-35812147-4)
O_{14} =(I14,atom, lbhidalgo@clubhindu.com.ar)

Conjuntos

Sedes
O_{15} =(I15, set , { I3, I4 , I5 })

Deportes
O_{16} =(I16, set , { I6, I7 , I8 })

Tuplas

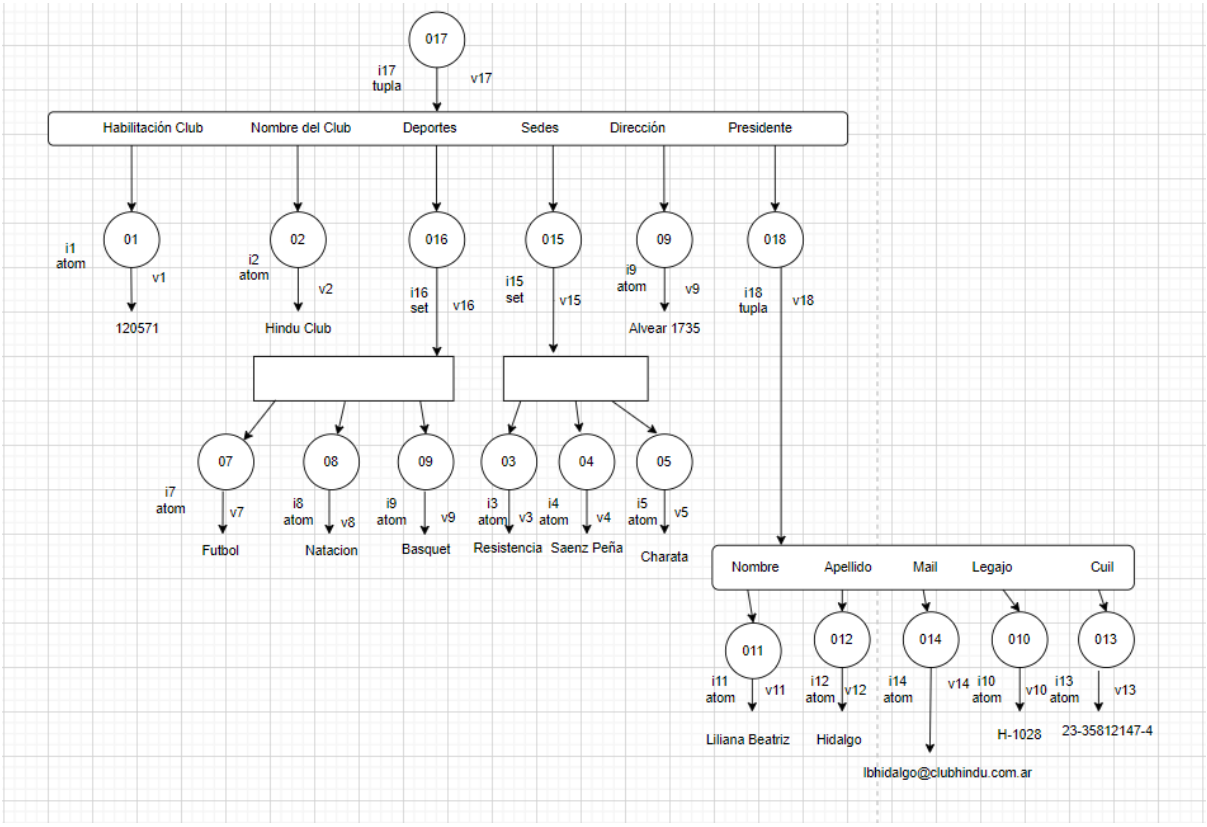
Club deportivo

O_{17} =(I17,tupla,< Habilitación Club : I1, Nombre Club : I2, Deportes : I16, Sedes: I15 ,Dirección: I9, Presidente I18>)

Presidente

O_{18} =(I18,tupla,< Nombre : I11, Apellido : I12 ,Legajo: I10 , Cuil : I13>)

8-b)



1)
a)

```
Create table Libros (
        Cod_libro integer generated always as identify (
                start with 1
                increment by 1
                minvalue 1
                maxvalue 5000
                no cycle),
        Titulo   varchar(40),
        Autor    varchar(35),
        Año      char(4),
        Precio   decimal(10,2)
);
```

b)

```
Create table Editorial (
        Cod_libro        integer
        Titulo   varchar(50),
        Descripcion      clob(40k),
        Autor    varchar(40),
        Foto_portada     blob(2m),
        Videopresen      blob(1G)
);
```

2)

```
Create type persona as (
        Dni      char(8),
        Apeynom          varchar2(50),
        Direccion        varchar2(40),
        Ciudad char(25),
        Fechanac date) not final;

Create type empresa as (
        Empleado         persona,
        Mail     varchar2(50),
        Codpostal         integer) not final;

Create type jefe as (
        Director         persona,
        Mail     varchar2(50),
        Area     char(2),
        Sueldo  decimal(8,2)) final;

Create table empleados of empresa;
```

3) a)

```
Create table Universidad (
        Nombre           varchar2(100),
        Rector  varchar2(50),
        Direccion        varchar(40),
        Facultades       varchar(15) array[9],
        Codpostal         integer,
        Sitio_web        varchar2(80));
```

Variante: para definir facultad de tipo multiconjunto.

Facultades varchar(15) multiset

3) b)

Insert into universidad (nombre, rector, direccion, facultades, codpostal, sitio_web)
Values (‘Universidad Nacional del Nordeste’,
‘Maria Veirave’,
’25 de Mayo 868’,
[‘Medicina’, ‘Humanidades’, ‘Exactas’, ‘Veterinaria’, ‘Económicas’,
‘Agrarias’, ‘Arquitectura’, ‘Derecho’, ‘Odontología’],
3400, ‘www.unne.edu.ar’);

Variante: para incorporar valores en el atributo facultad de tipo multiconjunto:
Insert into universidad (nombre, rector, direccion, facultades, codpostal, sitio_web)
Values (‘Universidad Nacional del Nordeste’,
‘Maria Veirave’,
’25 de Mayo 868’,
Multiset[‘Medicina’, ‘Humanidades’, ‘Exactas’, ‘Veterinaria’, ‘Económicas’,
‘Agrarias’, ‘Arquitectura’, ‘Derecho’, ‘Odontología’],
3400, ‘www.unne.edu.ar’);

4)

```
CREATE TABLE Banco(
        Identificación row (
        Nro_BCRA INTEGER,
        Cuit VARCHAR(13)

      Razon_social row(
      Nombre_comercial  VARCHAR2(100),
      Tipo_empresa  INTEGER(2),
      Condición_AFIP  CHAR )

      Presidente row(
      Nombres VARCHAR2(50),
      Apellido VARCHAR2(50),
      DNI CHAR(8),
      Mail VARCHAR(80) )

      Direccion row(
      Calle VARCHAR2(100),
      Numero INTEGER,
      Ciudad VARCHAR2(60),
      Cod_postal INTEGER )

      Telefono row(
      Prefijo INTEGER,
      Numero INTEGER )

      Sitio_web VARCHAR2(100);
);
```

5)

```
/* Creamos el tipo Empleado con sus correspondientes atributos */
CREATE TYPE Empleado as(
        DNI     CHAR(8),
        ApeyNom         VARCHAR2(100),
        Method Antigüedad_lab() returns INTEGER,
```

```
        Direccion       VARCHAR2(100),
        Cargo   VARCHAR2(30),
        Method sueldo() returns DECIMAL(8,2)

/*Creamos  el método de Antigüedad que se obtiene restando el año actual menos el año de
ingreso del empleado */
CREATE METHOD Antigüedad_lab() FOR Empleado
        Begin
                Return(Año-actual – Año-ingreso);
        End;

/* Creamos el método para obtener el sueldo del empleado sumando el sueldo básico junto con
el adicional por título y la escolaridad. Luego se resta el aporte jubilatorio*/
CREATE METHOD sueldo() FOR Empleado
        Begin
                Return(básico  +  adicional-titulo + escolaridad – aporte-jubilatorio);
        End;
);
```

6)

```
/* Creamos el tipo de dato Producto con sus atributos */
CREATE TYPE Producto as(
        Código_prod INTEGER,
        Denominación VARCHAR2(80),
        Stock_actual INTEGER,
        Stock_minimo INTEGER,
        Precio_fabrica DECIMAL(8,2),
        Method precioConsumidor() returns DECIMAL(8,2)
        );

        /* Creamos el método para obtener el precio consumidor.*/
        CREATE METHOD precioConsumidor() FOR Producto
                Begin
                        Return(precio-fabrica  + (precio-fabrica * 0,15))
                End;
)
```

7)

```
/* Creamos un tipo de datos estructurado llamado dirección_postal */
CREATE TYPE direccion_postal  as(
        Calle    VARCHAR2(80),
        Numero          INTEGER,
        Provincia       VARCHAR2(50),
        Cod_postal       INTEGER
);

/* Creamos una tabla Personas y en el atributo “direccion” utilizamos el tipo de dato creado
anteriormente */
CREATE TABLE Personas(
        DNI     CHAR(8) PRIMARY KEY,
        Nombre                  VARCHAR2(50)
        Apellidos       VARCHAR2(70),
        Fecha_Nac       DATE,
        Telefonos       TEXT[],
        Dirección       direccion_postal
);
```