UNIVERSIDAD NACIONAL DE LA MATANZA

DEPARTAMENTO DE INGENIERÍA
E INVESTIGACIONES TECNOLÓGICAS

INGENIERIA EN INFORMATICA

BASE DE DATOS

Ejercicios Propuestos y Resueltos
ANSI SQL

Jefe de Cátedra: Ing. Verónica Ichazo

Docentes a cargo de curso:
Ing. Alfonso Palomares
Ing. Natalia Crespo

Docentes a cargo de práctica:
Ing. Matías López
Ing. Hernán Jalil
Ing. Fernando Ybarra

2024

EJERCICIO 1

Dada la siguiente base de datos:

Almacén (**Nro**, Responsable)
Artículo (**CodArt**, descripción, Precio)
Material (**CodMat**, Descripción)
Proveedor (**CodProv**, Nombre, Domicilio, Ciudad)
Tiene (**Nro, CodArt**)
Compuesto_por (**CodArt, CodMat**)
Provisto_por (**CodMat, CodProv**)

Creación de Tablas:

```
create table Almacen(Nro int primary key, Responsable varchar(50))
create table Articulo(CodArt int primary key, Descripcion varchar(50), Precio
decimal(12, 3))
create table Material(CodMat int primary key, Descripcion varchar(50))
create table Proveedor(CodProv int primary key, Nombre varchar(50), Domicilio
varchar(50), Ciudad varchar(50))
create table Tiene(Nro int, CodArt int)
create table Compuesto_Por(CodArt int, CodMat int)
create table Provisto_Por(CodMat int, CodProv int)
```

Inserción de Datos

```
insert into Almacen values
  (1, 'Juan Perez'),
  (2, 'Jose Basualdo'),
  (3, 'Rogelio Rodriguez')

insert into Articulo values
  (1, 'Sandwich JyQ', 5),
  (2, 'Pancho', 6),
  (3, 'Hamburguesa', 10),
  (4, 'Hamburguesa completa', 15)

insert into Material values
  (1, 'Pan'),
  (2, 'Jamon'),
  (3, 'Queso'),
  (4, 'Salchicha'),
  (5, 'Pan Pancho'),
  (6, 'Paty'),
  (7, 'Lechuga'),
  (8, 'Tomate')

insert into Proveedor values
  (1, 'Panadería Carlitos', 'Carlos Calvo 1212', 'CABA'),
  (2, 'Fiambres Perez', 'San Martin 121', 'Pergamino'),
  (3, 'Almacen San Pedrito', 'San Pedrito 1244', 'CABA'),
  (4, 'Carnicería Boedo', 'Av. Boedo 3232', 'CABA'),
  (5, 'Verdulería Platense', '5 3232', 'La Plata')

insert into Tiene values
  --Juan Perez
  (1, 1),
  --Jose Basualdo
  (2, 1),
  (2, 2),
  (2, 3),
  (2, 4),
```

```
  --Rogelio Rodriguez
  (3, 3),
  (3, 4)

insert into Compuesto_Por values
  --Sandwich JyQ
  (1, 1), (1, 2), (1, 3),
  --Pancho
  (2, 4), (2, 5),
  --Hamburguesa
  (3, 1), (3, 6),
  --Hamburguesa completa
  (4, 1), (4, 6), (4, 7), (4, 8)

insert into Provisto_Por values
  --Pan
  (1, 1), (1, 3),
  --Jamon
  (2, 2), (2, 3), (2, 4),
  --Queso
  (3, 2), (3, 3),
  --Salchicha
  (4, 3), (4, 4),
  --Pan Pancho
  (5, 1), (5, 3),
  --Paty
  (6, 3), (6, 4),
  --Lechuga
  (7, 3), (7, 5),
  --Tomate
  (8, 3), (8, 5)
GO
```

1. Listar los nombres de los proveedores de la ciudad de La Plata.

   ```
   SELECT Nombre
   FROM Proveedor
   WHERE Ciudad = 'La Plata'
   ```

2. Listar los números de artículos cuyo precio sea inferior a $10.

   ```
   SELECT CodArt
   FROM Articulo
   WHERE Precio<10
   ```

3. Listar los responsables de los almacenes.

   ```
   SELECT Responsable
   FROM Almacen
   ```

4. Listar los códigos de los materiales que provea el proveedor 3 y no los provea el proveedor 5.

   ```
   SELECT CodMat
   FROM Provisto_por
   WHERE CodProv = 3 AND CodMat NOT IN (SELECT CodMat
                                        FROM Provisto_por
                                        WHERE CodProv = 5)
   ```

5. Listar los números de almacenes que almacenan el artículo 1.

```
SELECT Nro
FROM Tiene
WHERE CodArt = 1
```

6. Listar los proveedores de Pergamino que se llamen Pérez.

```
SELECT CodProv
FROM Proveedor
WHERE Ciudad = 'Pergamino' AND Nombre LIKE '%Perez'
```

7. Listar los almacenes que contienen los artículos 1 y los artículos 2 (ambos).

```
SELECT Nro
FROM Tiene
WHERE CodArt = 'B' AND Nro IN (SELECT Nro
                               FROM Tiene
                               WHERE CodArt = 'A')
```

8. Listar los artículos que cuesten más de $100 o que estén compuestos por el material 1.

```
SELECT A.CodArt
FROM Articulo A JOIN Compuesto_por C ON A.CodArt = C.CodArt
WHERE A.Precio>100 AND C.CodMat = M1
```

EJERCICIO 2

Dada la siguiente base de datos:

Almacén (**Nro**, Responsable)
Artículo (**CodArt**, descripción, Precio)
Material (**CodMat**, Descripción)
Proveedor (**CodProv**, Nombre, Domicilio, Ciudad)
Tiene (**Nro, CodArt**)
Compuesto_por (**CodArt, CodMat**)
Provisto_por (**CodMat, CodProv**)

Scripts Creación de Tablas:

```
create table Almacen(Nro int primary key, Responsable varchar(50))
create table Articulo(CodArt int primary key, Descripcion varchar(50), Precio decimal(12, 3))
create table Material(CodMat int primary key, Descripcion varchar(50))
create table Proveedor(CodProv int primary key, Nombre varchar(50), Domicilio varchar(50),
Ciudad varchar(50))
create table Tiene(Nro int, CodArt int)
create table Compuesto_Por(CodArt int, CodMat int)
create table Provisto_Por(CodMat int, CodProv int)
GO
```

Scripts Inserción de Datos

```
insert into Almacen values
 (1, 'Juan Perez'),
 (2, 'Jose Basualdo'),
 (3, 'Rogelio Rodriguez')

insert into Articulo values
 (1, 'Sandwich JyQ', 5),
 (2, 'Pancho', 6),
 (3, 'Hamburguesa', 10),
 (4, 'Hamburguesa completa', 15)

insert into Material values
 (1, 'Pan'),
 (2, 'Jamon'),
 (3, 'Queso'),
 (4, 'Salchicha'),
 (5, 'Pan Pancho'),
 (6, 'Paty'),
 (7, 'Lechuga'),
 (8, 'Tomate')

insert into Proveedor values
 (1, 'Panadería Carlitos', 'Carlos Calvo 1212', 'CABA'),
 (2, 'Fiambres Perez', 'San Martin 121', 'Pergamino'),
 (3, 'Almacen San Pedrito', 'San Pedrito 1244', 'CABA'),
 (4, 'Carnicería Boedo', 'Av. Boedo 3232', 'CABA'),
 (5, 'Verdulería Platense', '5 3232', 'La Plata')

insert into Tiene values
 --Juan Perez
 (1, 1),
 --Jose Basualdo
 (2, 1),
 (2, 2),
```

```
 (2, 3),
 (2, 4),
 --Rogelio Rodriguez
 (3, 3),
 (3, 4)

insert into Compuesto_Por values
 --Sandwich JyQ
 (1, 1), (1, 2), (1, 3),
 --Pancho
 (2, 4), (2, 5),
 --Hamburguesa
 (3, 1), (3, 6),
 --Hamburguesa completa
 (4, 1), (4, 6), (4, 7), (4, 8)

insert into Provisto_Por values
 --Pan
 (1, 1), (1, 3),
 --Jamon
 (2, 2), (2, 3), (2, 4),
 --Queso
 (3, 2), (3, 3),
 --Salchicha
 (4, 3), (4, 4),
 --Pan Pancho
 (5, 1), (5, 3),
 --Paty
 (6, 3), (6, 4),
 --Lechuga
 (7, 3), (7, 5),
 --Tomate
 (8, 3), (8, 5)
GO
```

1. Listar los nombres de los proveedores de la ciudad de La Plata.

Resultado esperado:

NOMBRE
Verdulería Platense

Solución:

```
SELECT Nombre
FROM Proveedor
WHERE Ciudad = 'La Plata'
```

2. Listar los números de artículos cuyo precio sea inferior a $10.

Resultado esperado:

CODART
1
2

Solución:

```
SELECT CodArt
FROM Articulo
WHERE Precio<10
```

3. Listar los responsables de los almacenes.

Resultado esperado:

RESPONSABLE
Juan Perez
Jose Basualdo
Rogelio Rodriguez

Solución:

```
SELECT Responsable
FROM Almacen
```

4. Listar los códigos de los materiales que provea el proveedor 3 y no los provea el proveedor 5.

Resultado esperado:

CODMAT
1
2
3
4
5
6

Solución:

```
SELECT CodMat
FROM Provisto_por
WHERE CodProv = 3 AND CodMat NOT IN (SELECT CodMat
   FROM Provisto_por
   WHERE CodProv = 5)
```

5. Listar los números de almacenes que almacenan el artículo 1.

Resultado esperado:

NRO
1
2

Solución:

```
SELECT Nro
FROM Tiene
WHERE CodArt = 1
```

6. Listar los proveedores de Pergamino que se llamen Pérez

Resultado esperado:

CODPROV
2

Solución:

```
SELECT CodProv
FROM Proveedor
WHERE Ciudad = 'Pergamino' AND Nombre like '% Perez'
```

7. Listar los almacenes que contienen los artículos 1 y los artículos 2 (ambos).

Resultado esperado:

NRO
2

Solución:

```
SELECT Nro
FROM Tiene
WHERE CodArt = 2
AND Nro IN (SELECT Nro
       FROM Tiene
       WHERE CodArt = 1)
```

8. Listar los artículos que cuesten más de $100 o que estén compuestos por el material 1.

Resultado esperado:

CODART
1
3
4

Solución:

```
SELECT DISTINCT A.CodArt
FROM Articulo A JOIN Compuesto_por C ON A.CodArt = C.CodArt
WHERE A.Precio>100 OR C.CodMat = 1
```

EJERCICIO 3

Dado el siguiente esquema de relación:

ALUMNO(**dni**, apellido, nombre, escuela)
HERMANO_DE(**dniAlum, dniHno**)
ESCUELA (**cod**, nombre, direccion)
ALIMENTO(**id**, descripcion, marca)
ALMUERZA_EN(**dniAlum, idAlimento**, codEscuela)

a) Listar a todos los alumnos que asisten a escuelas donde no sirven alimentos y almuerzan en otro establecimiento
b) Mostrar todas las escuelas que sirven alimentos a todos sus alumnos que no tienen más de dos hermanos

Creación de Tablas

```
create table Alumno(DNI int not null primary key, Apellido varchar(50), Nombre
varchar(50), CodEscuela int);
create table Hermano_De(DniAlumno int not null, DniHermano int not null, constraint
PK_Hermano_De primary key(DniAlumno, DniHermano));
create table Escuela(CodEscuela int not null primary key, Nombre varchar(50), Direccion
varchar(255));
create table Alimento(IdAlimento int not null primary key, Descripcion varchar(50),
Marca varchar(50));
create table Almuerza_En(DniAlumno int not null, IdAlimento int not null, CodEscuela
int, constraint PK_Almuerza_En primary key(DniAlumno, IdAlimento));
```

Inserción de Datos

```
INSERT INTO escuela (CodEscuela, Nombre, Direccion) VALUES ('1', 'Escuela 1', 'Famosos');
INSERT INTO escuela (CodEscuela, Nombre, Direccion) VALUES ('2', 'Escuela 2', 'Oficialistas');
INSERT INTO escuela (CodEscuela, Nombre, Direccion) VALUES ('3', 'Escuela 3', 'Opositores');
INSERT INTO escuela (CodEscuela, Nombre, Direccion) VALUES ('4', 'Escuela 4', 'Hermanos');



INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('1', 'Fort', 'Ricardo', '1');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('2', 'Marcelo', 'Tinelli', '1');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('3', 'Moria', 'Casan', '1');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('4', 'Cristina', 'Fernandez',
'2');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('5', 'Anibal', 'Fernandez',
'2');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('6', 'Amado', 'Boudou', '2');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('7', 'Ricardo', 'Alfonsin',
'3');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('8', 'Elisa', 'Carrio', '3');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('9', 'Hermes', 'Binner', '3');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('10', 'Guido', 'Tinelli', '4');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('11', 'Hugo', 'Tinelli', '4');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('12', 'Alberto', 'Fernandez',
'4');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('13', 'Silvia', 'Fernandez',
'4');
INSERT INTO alumno(DNI, Apellido, Nombre, CodEscuela) VALUES ('14', 'Ricardo', 'Tinelli',
'4');


INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('2', '10');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('2', '11');
```

```
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('2', '14');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('5', '12');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('4', '13');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('10', '2');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('10', '11');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('10', '14');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('11', '2');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('11', '10');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('11', '14');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('14', '2');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('14', '10');
INSERT INTO hermano_de (DniAlumno, DniHermano) VALUES ('14', '11');

INSERT INTO alimento (IdAlimento, Descripcion, Marca) VALUES ('1', 'Hamburguesa', 'Patty');
INSERT INTO alimento (IdAlimento, Descripcion, Marca) VALUES ('2', 'Milanesa', 'Granja del
Sol');
INSERT INTO alimento (IdAlimento, Descripcion, Marca) VALUES ('3', 'Salchicha', 'Vienisima');

INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('4', '1', '1');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('5', '1', '3');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('4', '2', '4');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('1', '3', '1');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('1', '1', '4');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('2', '1', '1');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('3', '1', '1');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('12', '2', '4');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('13', '2', '4');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('10', '1', '3');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('7', '1', '3');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('8', '2', '3');
INSERT INTO almuerza_en (DniAlumno, IdAlimento, CodEscuela) VALUES ('9', '3', '3');
```

a)

Resultado esperado:

DNI	Apellido	Nombre	CodEscuela
4	Cristina	Fernandez	2
5	Anibal	Fernandez	2

Solución:

```
/* (1) Escuelas que no sirven alimentos */
select *
from Escuela
where CodEscuela not in (
      select CodEscuela
      from almuerza_en
);

/* (1b) Alumnos que asisten a las escuelas de (1) */
select *
from Alumno A
where CodEscuela not in (
      select CodEscuela
      from almuerza_en
);

/* (2) Alumnos que almuerzan en otro establecimiento */
select *
from Alumno A
join Almuerza_En AE on A.DNI = AE.DniAlumno
```

```
where A.CodEscuela <> AE.CodEscuela;

/* Respuesta - (1b) y (2) */
select distinct A.*
from Alumno A
join Almuerza_En AE on A.DNI = AE.DniAlumno
where A.CodEscuela <> AE.CodEscuela
and A.CodEscuela not in (
      select CodEscuela
      from almuerza_en
);
```

b)*Resultado esperado:*

CodEscuela
3

Solución:

```
/* (1) Alumnos que tienen mas de dos hermanos */
select DniAlumno
from Hermano_De
group by DniAlumno
having count(1) > 2;

/* (2) Alumnos que no tienen mas de dos hermanos */
select *
from Alumno
where DNI not in (
      select DniAlumno
      from Hermano_De
      group by DniAlumno
      having count(1) > 2
);

/* (3) Escuelas que no le dan alimento a alguno de sus alumnos de (2) */
select *
from Alumno E1
where not exists(
      select 1
      from Almuerza_En E2
      join Alumno A on A.DNI = E2.DniAlumno
      where DNI not in (
            select DniAlumno
            from Hermano_De
            group by DniAlumno
            having count(1) > 2
      )
      and E1.DNI = E2.DniAlumno
      and E1.CodEscuela = E2.CodEscuela
);

/* (4) Escuelas que dan alimento a algún alumno */
select distinct CodEscuela
from almuerza_en;

/* Resultado: (4) - (3) */
```

```
select distinct CodEscuela
from almuerza_en
where CodEscuela not in (
      select distinct E1.CodEscuela
      from Alumno E1
      where not exists(
            select 1
            from Almuerza_En E2
            join Alumno A on A.DNI = E2.DniAlumno
            where DNI not in (
                  select DniAlumno
                  from Hermano_De
                  group by DniAlumno
                  having count(1) > 2
            )
            and E1.DNI = E2.DniAlumno
            and E1.CodEscuela = E2.CodEscuela
      )
);
```

EJERCICIO 4

BANCO (**id**, nombre, pais)
CUENTA (monto, ***idBanco, idMoneda, idPersona***)
MONEDA (**id**, descripción, valor_oro, valor_petroleo)
OPERA (***idBanco, idMoneda***, cambio_compra, cambio_venta)
PAIS (nombre)
PERSONA(**pasaporte**, codigofiscal, nombre)

Todas las operaciones realizadas para los cambios de moneda hacen referencia al dólar. Es decir, para convertir pesos a euros, debo convertir pesos a dólares, luego dolares a euros.

A. Listar a las personas que no tienen ninguna cuenta en ***"pesos argentinos"*** en Ningún banco. Que además tengan al menos dos cuentas en ***"dólares"***

B. Listar de las monedas que son operadas en todos los bancos, aquellas con el valor oro más alto.

Creación de Tablas

```
create table Pais(pais char(50) primary key);
create table Banco(id int primary key, nombre varchar(50), pais char(50));
create table Moneda(id char(2) primary key, descripcion varchar(50), valorOro
decimal(18,3), valorPetroleo decimal(18,3));
create table Persona(pasaporte char(15) primary key, codigoFiscal int, nombre
varchar(50));
create table Cuenta(monto decimal(18,3), idBanco int not null, idMoneda char(2) not
null, idPersona char(15) not null, constraint PK_Persona primary key(idBanco, idMoneda,
idPersona));
create table Opera(idBanco int not null, idMoneda char(2) not null, cambioCompra
decimal(18,3), cambioVenta decimal(18,3), constraint PK_Opera primary key(idBanco,
idMoneda));
```

Inserción de datos:

```
INSERT INTO pais (pais) VALUES ('Argentina');
INSERT INTO pais (pais) VALUES ('USA');
INSERT INTO pais (pais) VALUES ('Uruguay');
INSERT INTO pais (pais) VALUES ('España');
INSERT INTO pais (pais) VALUES ('Alemania');
INSERT INTO pais (pais) VALUES ('Suiza');

INSERT INTO banco (id, nombre, pais) VALUES ('1', 'Banco Nacion', 'Argentina');
INSERT INTO banco (id, nombre, pais) VALUES ('2', 'Banco Montevideo', 'Uruguay');
INSERT INTO banco (id, nombre, pais) VALUES ('3', 'Banco Ciudad', 'Argentina');
INSERT INTO banco (id, nombre, pais) VALUES ('4', 'City Bank', 'USA');
INSERT INTO banco (id, nombre, pais) VALUES ('5', 'Switzerland Bank', 'Suiza');
INSERT INTO banco (id, nombre, pais) VALUES ('6', 'BBVA', 'España');

INSERT INTO moneda (id, descripcion, valorOro, valorPetroleo) VALUES ('AR', 'Peso Argentino',
'2', '1');
INSERT INTO moneda (id, descripcion, valorOro, valorPetroleo) VALUES ('UY', 'Peso Uruguayo',
'5', '2.5');
INSERT INTO moneda (id, descripcion, valorOro, valorPetroleo) VALUES ('US', 'Dolar', '1',
'.5');
INSERT INTO moneda (id, descripcion, valorOro, valorPetroleo) VALUES ('EU', 'Euro', '2', '1');

INSERT INTO persona (pasaporte, codigoFiscal, nombre) VALUES ('1', '1234', 'Bill Gates');
INSERT INTO persona (pasaporte, codigoFiscal, nombre) VALUES ('2', '12112', 'Carlos Slim');
INSERT INTO persona (pasaporte, codigoFiscal, nombre) VALUES ('3', '2325', 'Lionel Messi');
INSERT INTO persona (pasaporte, codigoFiscal, nombre) VALUES ('4', '01243', 'Diego Maradona');

INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('100000', '4', 'US', '1');
```

```
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('20000', '5', 'EU', '1');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('15000', '2', 'US', '1');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('50000', '4', 'US', '2');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('35000', '5', 'US', '2');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('2000', '1', 'AR', '3');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('10000', '4', 'US', '3');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('15000', '5', 'US', '3');
INSERT INTO cuenta (monto, idBanco, idMoneda, idPersona) VALUES ('15000', '5', 'US', '4');

INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('1', 'US', '1', '1');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('2', 'US', '1', '1');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('3', 'US', '1', '1');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('4', 'US', '1', '1');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('5', 'US', '1', '1');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('6', 'US', '1', '1');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('1', 'EU', '2', '2');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('2', 'EU', '2', '2');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('3', 'EU', '3', '3');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('4', 'EU', '2', '2');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('5', 'EU', '2.2',
'2.2');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('6', 'EU', '2.2',
'2.5');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('1', 'AR', '5', '5');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('3', 'AR', '5.5',
'5.5');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('2', 'AR', '7', '7');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('1', 'UY', '3', '3');
INSERT INTO opera (idBanco, idMoneda, cambioCompra, cambioVenta) VALUES ('2', 'UY', '2', '2');
```

a)

Resultado esperado:

Pasaporte	CodigoFiscal	Nombre
1	1234	Bill Gates
2	12112	Carlos Slim

Solución:

```
/*Personas con cuentas en Pesos*/
	SELECT idPersona
	FROM cuenta c JOIN moneda m ON m.id = c.idmoneda
	WHERE m.descripcion = 'Peso Argentino';

/*Personas que tienen mas de dos cuentas en dólares*/
	SELECT idpersona
	FROM cuenta c JOIN moneda m ON m.id = c.idmoneda
	WHERE m.descripcion = 'dolar'
	GROUP by idpersona
	HAVING COUNT(distinct idbanco)>=2;

/*Solución*/
	SELECT *
	FROM persona P1
	WHERE p1.pasaporte NOT IN
	(
		SELECT idPersona
		FROM cuenta c JOIN moneda m ON m.id = c.idmoneda
		WHERE m.descripcion = 'Peso Argentino'
	)
```

```
	AND EXISTS
	(
	SELECT 1
		FROM cuenta c JOIN moneda m ON m.id = c.idmoneda
		WHERE c.idpersona=p1.pasaporte
		AND m.descripcion = 'dolar'
		GROUP by idpersona
		HAVING COUNT(distinct idbanco)>=2
	);
```

b)

Resultado esperado:

Id	Descripcion
EU	Euro

Solución:

```
/*Monedas que están en todos los bancos*/

	SELECT *
	FROM moneda m
	WHERE NOT EXISTS
	(
		SELECT 1
		FROM banco b
		WHERE NOT EXISTS
		(
			SELECT 1 FROM opera o
			WHERE o.idmoneda = m.id AND o.idbanco = b.id
		)
	);

/*Solución*/

	CREATE VIEW monedas_en_todos_los_bancos AS
	SELECT *
	FROM moneda m
	WHERE NOT EXISTS
	(
		SELECT 1
		FROM banco b
		WHERE NOT EXISTS
		(
			SELECT 1 FROM opera o
			WHERE o.idmoneda = m.id AND o.idbanco = b.id
		)
	);

select id, descripcion from monedas_en_todos_los_bancos
where valororo >=
(
	select max(valororo) from monedas_en_todos_los_bancos
);
```

EJERCICIO 5

GaleríaDeArte(**id**, nombre, disponible, calle, nro, localidad)

Obra(**id**, nombre, material, idTipo, idAutor)

TipoDeObra(**id**, descripcion)

Temática(**id**, descripcion)

Exposición(**idGaleria, idObra, idTematica, fecha**, sala)

Autor(**id**, nya, fechaNacimiento)

-- Creación de estructuras.

```
CREATE TABLE GaleriaDeArte
(id int primary key, nombre varchar(50), disponible varchar(50), calle varchar(50), nro
varchar(50), localidad varchar(50));

CREATE TABLE Autor
(id INT PRIMARY KEY, nya varchar(50), fech_nacimiento DATE );

CREATE TABLE TipoDeObra
(id int primary key, descripcion varchar(50));


CREATE TABLE Obra
(id int primary key, nombre varchar(50), material varchar(50),
idTipo int, idAutor int,
FOREIGN KEY (idTipo ) REFERENCES TipoDeObra(id),
FOREIGN KEY (idAutor ) REFERENCES Autor(id) );


CREATE TABLE Tematica
(id int primary key, descripción varchar(50));


CREATE TABLE Exposicion
(idGaleria int , idObra int , idTematica int , fecha date, sala int,
PRIMARY KEY(idGaleria, idObra, idTematica, fecha),
FOREIGN KEY (idGaleria ) REFERENCES GaleriaDeArte (id),
FOREIGN KEY (idObra ) REFERENCES TipoDeObra(id),
FOREIGN KEY (idTematica ) REFERENCES Tematica(id)
);
```

-- Insertando datos.

```
INSERT INTO GaleriaDeArte VALUES
(1, 'Galeria barcelona', '', '', '',''),
(2, 'Galeria buenos aires', '', '', '',''),
(3, 'Galeria Florencia', '', '', '',''),
(4, 'Galeria Recoleta', '', '', '',''),
(5, 'Galeria Orfeo', '', '', '','');

INSERT INTO Autor VALUES
(1, 'dali', '1904-05-11'),
(2, 'picasso', '1881-10-25'),
```

```
(3, 'Joan Miro', '1893-04-20'),
(4, 'Max Ernst', '1891-04-02'),
(5, 'Man Ray', '1890-08-27');

INSERT INTO TipoDeObra VALUES
(1,'dadaísmo'),
(2,'surrealismo'),
(3,'pop art'),
(4,'Art Deco'),
(5,'Minimalismo');


INSERT INTO OBRA VALUES
(),
(),
(),
(),
();

INSERT INTO Tematica VALUES
(),
(),
(),
(),
();

INSERT INTO Exposicion VALUES
(),
(),
(),
(),
();
```

a - Obtener el nombre de la galería de arte, la descripción de la temática presentada y la fecha de realización, cuando la exposición tuvo la mayor cantidad de obras en expuestas. Sólo se mostrarán los resultados siempre y cuando la galería de arte haya presentado todas las temáticas disponibles o haya expuesto distintas obras a tal punto de haber presentado todos los tipos de obra disponibles.

```
create view exposicion_cantidadObras
as
select distinct e.id_galeria, e.id_tematica, e.fecha, count(id_obra) as cantObras
from exposicion e
group by e.id_galeria, e.id_tematica, e.fecha

--

create view exposicion_obrasMax
as
select e.id_galeria, e.id_tematica, e.fecha
from exposicion_cantidadObras
where cantObras = (select max(cantObras) from exposicion_cantidadObras)
--
```

```
create view geleria_todasTematicas
as
select id_galeria, count(id_tematica) tematicasPresentadas
from exposicion
group by id_galeria
having tematicasPresentadas = (select count(*) from tematicas)

--

create view galeria_todosTiposObra
as
select id_galeria, count(o.id_tipo) cantidadTiposObra
from exposicion e
join obra o on e.id_obra = o.id
group by id_galeria
having cantidadTiposObra = (select count(*) from tipoDeObra)

--

select *
from exposicion_obrasMax
where id_galeria in (select id_galeria from galeria_todasTematicas)
or id_galeria in (select id_galeria from galeria_todosTiposObra)
```

b - Se requiere crear un procedimiento almacenados q o función (PostgreSQL) para generar una nueva exposición, por lo tanto se desea recibir por parámetro, el id de la galería de arte, id de la temática, id de la obra a participar y la fecha. Si la exposición no existe se deberá asignar el número de sala “1”, pero si la exposición ya existiera deberá utilizarse el número de sala previamente cargado para la misma.
Aclaración: Deberá validar que los id recibidos por parámetros existan en las tablas correspondientes.

-- LENGUAJE: SQL SERVER

```
CREATE PROCEDURE nueva_exposicion
@id_galeria as int,
@id_tematica as int,
@id_obra as int
@fecha as datetime
AS

declare @sala as int;


IF ( NOT EXISTS (SELECT 1 FROM GaleriaDeArte WHERE id = @id_galera)
      OR NOT EXISTS (SELECT 1 FROM Tematica WHERE id = @id_tematica)
      OR NOT EXISTS (SELECT 1 FROM Obra WHERE id = @id_obra)
      OR EXISTS (SELECT 1 FROM Exposicion WHERE id_obra = @id_obra AND id_tematica =
@id_tematica AND id_obra = @id_obra) )
BEGIN
```

```
      RETURN;
END


set @sala = SELECT sala from FROM Exposicion WHERE id_galeria = @id_galeria AND
id_tematica = @id_tematica AND fecha = @fecha;


IF @sala is null
BEGIN
      set @sala = 1;
END


INSERT INTO Exposicion (id_galeria, id_tematica, id_obra, fecha, sala) values
(@id_galeria, @id_tematica, @id_obra, @fecha, @sala)


GO
```