Valentin blanco, Pascual Coundannes, Sergio Calderon

**Bases De Datos 2022 Práctica 2- Álgebra relacional.**

**Consultas:**

|  |  |  |
| --- | --- | --- |
|  | Símbolo | Equivalencia SQL |
| Selección |  | **WHERE** |
| Proyección |  | **SELECT** |
| Producto Natural |  | **INNER JOIN … ON** |
| Producto Cartesiano |  | **FROM** |
| Renombre |  | **AS** |
| Unión |  |  |
| Intersección |  |  |
| Diferencia | - |  |
| Asignación |  |  |

**Altas, Bajas y Modificación:**

|  |  |
| --- | --- |
|  | Símbolo |
| Alta |  |
| Baja |  |
| Modificación |  |

**Glosario:**

Selección:

Proyección: π

Renombre: ρ

Producto cartesiano: x

Producto natural: |x|

Producto Theta:

Unión: È

Intersección: Ç

Diferencia: -

División: %

Asignación:

**1) Dadas las siguientes relaciones, resolver utilizando Álgebra Relacional las consultas planteadas.**

**Vuelo = (codVuelo,** fecha, hora, cod\_ciudad\_origen, cod\_ciudad\_destino, cantidad\_pasajes, codAerolinea**)**

**Pasaje = (codReserva,** asiento, codVuelo, codCliente, precio**)**

**Aerolinea = (codAerolinea,** nombre, origen**)**

**Cliente = (codCliente,** nombre, apellido, pasaporte, nacionalidad**)**

**Ciudad = (**codCiudad**,** nombre, pais**)** *// pais string con el nombre del país correspondiente*

**a. Listar datos personales de clientes que solo hayan viajado durante 2018.**

Primero se realiza el producto natural entre Vuelo y Pasaje para obtener también los códigos de clientes asociados a cada vuelo, y se filtran, por un lado, los de 2018, y por otro el resto de los vuelos registrados. Para obtener los datos personales, se realiza un producto natural entre Cliente y la tabla resultante de códigos “sólo 2018”, que sólo tiene codCliente.

Nota 1: NO se puede asumir que los clientes que sólo viajaron en 2018 son TODOS excepto los que viajaron hasta 2017 y desde 2019 (es decir, no usar el 1° filtro de fecha) ya que pueden existir clientes que están registrados pero nunca tomaron un vuelo.

Tabla 1 – Operaciones entre Vuelo y Pasaje

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | fecha | hora | codCO | codCD | cantP | codA | codR | asien | codC | prec |
| |X| |  |  |  |  |  |  |  |  |  |  |

Tabla 2 – Operaciones entre Cliente y Tabla 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| codCliente | nombre | apellido | pasaporte | nacionalidad |
| |X| |  |  |  |  |

(Cliente|x|( (pasaje vuelo))) - (Cliente|x|( (pasaje vuelo)))

**b. Borrar el vuelo con código de vuelo LOM3524.**

Para la baja se realiza una diferencia. Los operandos deben tener el mismo esquema, es decir, las columnas con el mismo orden y tipo. Esto se cumple al sólo realizar una selección.

Tabla 1 – Operaciones sobre Vuelo

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| codV | fecha | hora | codCO | codCD | cantP | codA |
|  | - | - | - | - | - | - |

Vuelo <-Vuelo -

**c. Listar datos personales de cliente que realizaron viajes con destino ‘Cancún’ durante 2018, pero no volaron durante 2019**

Tabla 1 – Operaciones entre Vuelo y Ciudad

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | fecha | hora | codCO | codCD | cantP | codA | codCI | nombre | pais |
|  |  |  |  |  |  |  |  |  |  |

Tabla 2 – Operaciones entre Vuelo, Pasaje y Cliente

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | codR | asiento | codCl | precio | nombre | apellido | pasap | nacion |
| |X| |  |  | |X| |  |  |  |  |  |

( (Cliente ( ( ( (Ciudad (( (pasaje vuelo))))))))) - (Cliente||( (pasaje vuelo)))

**d. Reportar información de vuelos con destino ‘Buenos Aires’ o que tengan pasajeros con nacionalidad ucraniana.**

- El producto cartesiano entre Vuelo y Ciudad mezcla cada vuelo con cada ciudad que existe.

- De todas las mezclas, me interesan las tuplas donde coincidan el código de la ciudad de destino con el código de la ciudad mezclada, de modo de obtener su nombre y país correspondiente.

- De todos los vuelos obtenidos, filtro los que tienen como ciudad destino a “Buenos Aires”.

Tabla 1 – Operaciones entre Vuelo y Ciudad

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | fecha | hora | codCO | codCD | cantP | codA | codCI | nombre | pais |
|  |  |  |  |  |  |  |  |  |  |

Tabla 2 – Operaciones entre Pasaje y Cliente

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codR | asiento | codV | codCl | precio | nombre | apellido | pasap | nacion |
|  |  |  | |X| |  |  |  |  |  |

( (Vuelo Ciudad) )) È ( ((Cliente||( pasaje vuelo)))))

È

**e. Listar datos personales de clientes que volaron con destino ‘Salta’ y también realizaron vuelos con destino ‘Jujuy’.**

Tabla 1 – Operaciones entre Vuelo y Ciudad

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | fecha | hora | codCO | codCD | cantP | codA | codCI | nombre | pais |
|  |  |  |  |  |  |  |  |  |  |

Tabla 2 – Operaciones entre Vuelo, Pasaje y Cliente

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | codR | asiento | codCl | precio | nombre | apellido | pasap | nacion |
| |X| |  |  | **|X|** |  |  |  |  |  |

( Ç)

))

**f. Listar información de aerolíneas que solo tengan vuelos con destino ‘Argentina’. Informar nombre y origen.**

- AeroArg me deja todas las aerolineas que vayan como mínimo a 1 ciudad de Argentina

- AeroNoArg me deja todas las aerolíneas que vayan como mínimo a 1 ciudad no argentina.

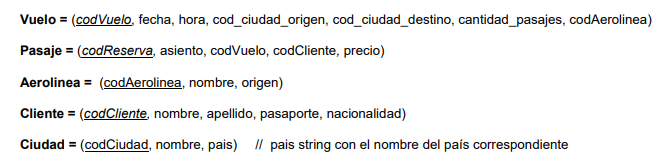
Tabla 1 – Operaciones entre Vuelo y Ciudad

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| codV | f | h | codCO | codCD | cantP | codA | codCI | C.nom | pais |
|  |  |  |  |  |  |  |  |  |  |

**g. Listar datos personales de clientes que viajaron con todas las aerolíneas.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| cCl | nom | ap | pas | nac | cR | as | cV | $ | f | h | cco | ccd | cp | cA |
| |X| |  |  |  |  |  |  | |X| |  |  |  |  |  |  | % |

((Cliente|x|(Pasaje|x|vuelo))% (Aerolínea))



**h. Lista la fecha, ciudad origen, ciudad destino, aerolínea y precio de los viajes del cliente con codigo 12345.**

Vuelo(Conecta todo), Aerolínea(nombre), Cliente(cod cliente), Ciudad (Para nombre), Pasaje (precio)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| cCl | nom | ap | pas | nac | cR | as | cV | $ | f | h | cco | ccd | cp | cA |
| |X| |  |  |  |  |  |  | |X| |  |  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| codVuelo | … | codCiudadOrigen | CO.codCiudad | CO.nombre | CO.país |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| codVuelo | … | codCiudadDest | CD.codCiudad | CD.nombre | CD.país |
|  |  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| codVuelo | … | codAerolinea | Aero.nombre | Aero.origen |
|  |  | |X| |  |  |

(())))

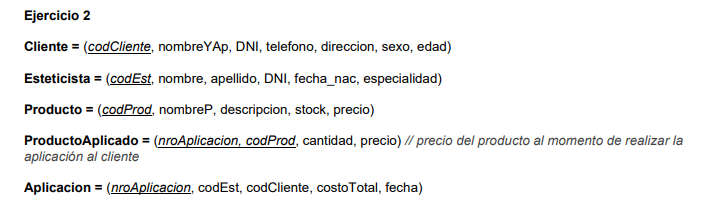
(

()

()

**i. Listar las ciudades que no son destino de ningún vuelo.**

Otra solución:



a. Listar datos personales de esteticistas que solo hayan atendido durante 2019.

b. Listar datos personales de esteticistas que no realizaron ninguna aplicación a clientes menores de 25 años.

c. Actualizar el precio de los productos de nombre ‘tintura’ incrementando 20% su valor actual.

d. Listar datos personales de clientes que se realizaron alguna aplicación durante 2018 pero no se atendieron en 2019.

e. Reportar nombre, descripción, stock y precio de productos que se utilizaron en mujeres y que tengan alguna aplicación donde el precio al momento de la aplicación fue inferior a $500.

f. Listar información de productos utilizados en las aplicaciones realizadas al cliente con DNI: 38329663.

g. Listar datos de productos utilizados por todos los esteticistas.

h. Listar los clientes que no tienen aplicaciones en el año actual.

i. Listar a los esteticistas Peinadores (especialidad), que no hayan realizado aplicaciones con un costo mayor a $2000.

**Bases De Datos-2022 Práctica 3- SQL**

**Ejercicios Dadas las siguientes relaciones, resolver en SQL las consultas planteadas:**

**Ejercicio 1**

**Cliente = (codCliente, nombreYAp, DNI, telefono, direccion, sexo, edad)**

**Esteticista = (codEst, nombre, apellido, DNI, fecha\_nac, especialidad)**

**Producto = (codProd, nombreP, descripcion, stock, precio)**

**ProductoAplicado = (nroAplicacion, codProd, cantidad, precio) // precio del producto al momento de realizar la aplicación al cliente**

**Aplicacion = (nroAplicacion, codEst, codCliente, costoTotal, fecha)**

**1. Listar datos personales de esteticistas que solo hayan atendido durante 2019, ordenar por nombre y apellidos y DNI ascendentemente.**

SELECT DISTINCT nombre, apellido, DNI, fecha\_nac, especialidad

FROM Esteticista AS E

INNER JOIN Aplicacion AS A ON (E.codEst=A.codEst)

WHERE (fecha BETWEEN “01/01/2019” and “31/12/2019”) and E.codEst NOT IN (

SELECT DISTINCT nombre, apellido, DNI, fecha\_nac, especialidad

FROM Esteticista AS E1

INNER JOIN Aplicacion AS A1 ON (E1.codEst=A1.codEst)

WHERE (YEAR(fecha)<>2019)

)

ORDER BY nombre, apellido, DNI

**2. Listar promedio de edad de clientes tratados con productos que terminen con el String ‘ura’.**

SELECT AVG(Edad) As Promedio

FROM Clientes AS C

INNER JOIN Aplicacion A ON (C.codCliente=A.codCliente)

INNER JOIN ProductoAplicado PA ON (A.nroAplicacion=PA.nroAplicacion)

INNER JOIN Producto P ON (PA.codProd=P.codProd)

WHERE (nombreP LIKE “%ura)

GROUPBY Producto

**3. Listar para cada producto, la cantidad de aplicaciones en las que fue utilizado. Indicar nombre, descripción, stock, precio y cantidad de aplicaciones. Ordenar por cantidad de aplicaciones.**

SELECT DISTINCT nombreP, descripción, stock, p.precio, COUNT(PA.nroAplicacion) AS CantAplicaciones

FROM Producto as P

LEFT JOIN ProductoAplicado PA ON (P.codProd = PA.codProd)

GROUP BY p.codProd, nombreP, descripcion, stock, p.precio

ORDER BY COUNT(PA.nroAplicacion)

**4. Listar datos personales de esteticistas que no realizaron ninguna aplicación a clientes menores de 25 años.**

SELECT nombre, apellido, DNI, fecha\_nac, especialidad

FROM Esteticista E

INNER JOIN Aplicación A ON (E.codEst=A.codEst)

INNER JOIN Cliente C ON (A.codCliente=C.codCliente)

WHERE (C.edad>=25) and C.codcliente NOT IN(

SELECT C1.codcliente

FROM cliente c1

WHERE (C1.edad<25)

**5. Actualizar el precio de los productos de nombre ‘tintura’ incrementando 20% su valor actual.**

UPDATE Producto

SET precio=precio\*1.2

WHERE (nombreP = “Tintura”)

**6. Listar datos personales de clientes que se realizaron alguna aplicación durante 2018 pero no se atendieron en 2019.**

SELECT nombreYAp, DNI, Telefono, dirección, sexo, edad

FROM Cliente C

INNER JOIN Aplicación as A ON (C.codCliente=A.conCliente)

WHERE (A.fecha BETWEEN “01/01/2018” and “31/12/2018”) and C.codCliente NOT IN(

SELECT A1.codCliente

From Aplicacion A1

WHERE (A1.fecha BETWEEN “01/01/2019” and “31/12/2019”)

)

**7. Reportar nombre, descripción,stock y precio de productos que se utilizaron en mujeres y que tengan alguna aplicación donde el precio al momento de la aplicación fue inferior a $500.**

SELECT nombre, descripción, stock, precio

FROM Producto P

INNER JOIN Aplicacion A ON (P.codProd=A.codProd)

INNER JOIN Cliente C ON (A.codCliente=C.codCliente)

WHERE (sexo=mujer) and A.costoTotal(

SELECT A1.costoTotal

FROM Aplicación A1

WHERE (costoTotal<=500)

)

**8. Listar información de productos utilizados en las aplicaciones realizadas al cliente con DNI: 38329663.**

SELECT DISTINCT codProd, nombreP, descripcion, stock, precio

FROM Producto P

INNER JOIN ProductoAplicado PA ON (P.codProd=PA.codProd)

INNER JOIN Aplicación A ON (PA.nroAplicacion=A.nroAplicacion)

INNER JOIN Cliente C ON (PA.codCliente=C.codCliente)

WHERE (DNI=38329663)

**9. Listar datos de productos utilizados por todos los esteticistas.**

SELECT Distinct nombreP, descripcion, stock, precio

FROM Producto as P

WHERE Not Exist (

(SELECT codEst

From Esteticista E)

EXCEPT

(SELECT codEst

FROM Aplicacion as A

INNER JOIN Aplicación A ON (PA.nroAplicacion=A.nroAplicacion)

WHERE (PA.codProd=p.codProd) )

**10. Listar el/los cliente que gastaron más en peluqueria (suma de costo total de sus aplicaciones).**

SELECT DISTINCT codCliente, nombreYAp, DNI, telefono, direccion, sexo, edad

FROM Cliente

NATURAL JOIN Aplicación as A

INNER JOIN Esteticista E ON (E.codEst=A.codEst)

WHERE(especialidad=“Peluqueria”)

GROUP BY codCliente, nombreYAp, DNI, telefono, direccion, sexo, edad

HAVING SUM(costoTotal) >= ALL (

SELECT SUM(costoTotal)

FROM Aplicacion

INNER JOIN Esteticista E ON (E.codEst=A.codEst)

WHERE(especialidad=“Peluqueria”)

GROUP BY codCliente)

**Ejercicio 2**

**Vuelo = (codVuelo, fecha, hora, cod\_ciudad\_origen, cod\_ciudad\_destino, cantidad\_pasajes, codAerolinea)**

**Pasaje = (codReserva, asiento, codVuelo, codCliente, precio)**

**Aerolinea = (codAerolinea, nombre, origen)**

**Cliente = (codCliente,** **nombre, apellido, pasaporte, nacionalidad)**

**Ciudad = (codCiudad, nombre, pais) // país string con el nombre del país correspondiente**

**1. Listar datos personales de clientes que viajaron con todas las aerolíneas.**

Select nombre, apellido, pasaporte, nacionalidad

From cliente c

Where not exist(

(Select codAerolinea

From Aerolínea

)except

(Select codaerolinea

From aerolínea a

Inner join vuelo v on (v.codAerolinea=a.codAerolinea)

Inner join pasaje p on (v.codvuelo=p.codvuelo)

Where (c.codCliente=P.codCliente)

)

)

**2. Reportar para cada cliente, la cantidad de vuelos realizados con destino buenos aires. Se debe informar datos personales del cliente y cantidad de vuelos.**

SELECT Distinct C.nombre, apellido, pasaporte, nacionalidad, Count(nombre) as cantidadVuelos

FROM Cliente C

INNER JOIN Pasaje P ON (C.codCliente=P.codCliente)

INNER JOIN Vuelo V ON (P.codVuelo=V.codVuelo)

INNER JOIN Ciudad Ciu ON (Ciu.codCiudad=V.cod\_ciudad\_destino)

WHERE (Ciu.nombre=“BuenosAires”)

Group By C.codCliente, C.nombre, apellido, pasaporte, nacionalidad

**3. Listar información de vuelos sobrevendidos, vendió más pasajes que los disponibles para el vuelo. Indicar codVuelo, fecha, hora, ciudad desde donde sale el vuelo y la ciudad donde llega.**

SELECT Distinct codVuelo, fecha, hora, Ciu1.nombre, Ciu2.nombre

FROM Vuelo as V

INNER JOIN Ciudad as Ciu1 ON (Ciu1.codCiudad=V.cod\_ciudad\_origen)

INNER JOIN Ciudad as Ciu2 ON (Ciu2.codCiudad=V.cod\_ciudad\_destino)

WHERE V.cantidad\_pasajes < (

SELECT COUNT(\*)

FROM Pasaje P

WHERE (P.codVuelo=V.codVuelo)

//GROUP BY P.codVuelo

)

**4. Listar datos personales de clientes que solo hayan viajado durante 2018. Ordenar por apellido y nombre**

SELECT Distinct nombre, apellido, pasaporte, nacionalidad

FROM Cliente C

INNER JOIN Pasaje P ON (P.codCliente=C.codCliente)

INNER JOIN Vuelo V ON (V.codVuelo=P.codVuelo)

WHERE (YEAR(V.fecha))=2018 and C.codCliente not in (

SELECT codCliente

FROM Pasaje P1

INNER JOIN Vuelo V1 ON (V1.codVuelo=P1.codVuelo)

WHERE (YEAR(V1.fecha))<>2018

)

**5. Listar para cada ciudad la cantidad de pasajeros que llegaron durante 2018. Indicar nombre, país y cantidad de pasajeros.**

SELECT nombre, país, SUM(cantidad\_pasajes) as CantidadPasajeros

FROM Ciudad Ciu

INNER JOIN Vuelo V ON (V.cod\_ciudad\_destino=Ciu.codCiudad)

WHERE (YEAR(V.fecha))=2018

GROUP BY codCiudad, nombre, país

**6. Borrar el vuelo con código de vuelo LOM3524.**

DELETE FROM Vuelo V

WHERE (codVuelo=“LOM3524”)

**7. Listar datos personales de cliente que realizaron viajes con destino ‘Cancún’ durante 2018, pero no volaron durante 2019**

SELECT Distinct nombre, apellido, pasaporte, nacionalidad

FROM Cliente C

INNER JOIN Pasaje P ON (P.codCliente=C.codCliente)

INNER JOIN Vuelo V ON (V.codVuelo=P.codVuelo)

INNER JOIN Ciudad Ciu ON (Ciu.codCiudad=V.cod\_ciudad\_destino)

WHERE (YEAR(V.fecha))=2018 and Ciu.nombre=“Cancún” and C.codCliente not in (

SELECT codCliente

FROM Pasaje P1

INNER JOIN Vuelo V1 ON (V1.codVuelo=P1.codVuelo)

WHERE (YEAR(V1.fecha))=2019

)

**8. Reportar información de vuelos con destino ‘Buenos Aires’ o que tengan pasajeros con nacionalidad ucraniana.**

SELECT Distinct codVuelo, fecha, hora, cod\_ciudad\_origen, cod\_ciudad\_destino, cantidad\_pasajes, codAerolinea

FROM Vuelo V

INNER JOIN Ciudad Ciu ON (V.cod\_ciudad\_destino=Ciu.codCiudad)

WHERE (Ciu.nombre=“Buenos Aires”)

UNION

SELECT Distinct codVuelo, fecha, hora, cod\_ciudad\_origen, cod\_ciudad\_destino, cantidad\_pasajes, codAerolinea

FROM Vuelo V

INNER JOIN Pasaje P ON (V.codVuelo=P.codVuelo)

INNER JOIN Cliente C ON (C.codCliente=P.codCliente)

WHERE (C.nacionalidad=“ucraniana”)

**9. Listar datos personales de clientes que volaron con destino ‘Salta’ y también realizaron vuelos con destino ‘Jujuy’.**

SELECT Distinct nombre, apellido, pasaporte, nacionalidad

From Cliente as C1

INNER JOIN Pasaje as P1 ON(C1.codCliente=P1.codCliente)

INNER JOIN Vuelo as V1 ON(P1.codVuelo=V1.codVuelo)

INNER JOIN Ciudad as Ciu1 ON(V1.cod\_ciudad\_destino=Ciu1.codCiudad)

WHERE (Ciu1.nombre=“Salta”)

INTERSECT

SELECT Distinct nombre, apellido, pasaporte, nacionalidad

From Cliente as C2

INNER JOIN Pasaje as P2 ON(C2.codCliente=P2.codCliente)

INNER JOIN Vuelo as V2 ON(P2.codVuelo=V2.codVuelo)

INNER JOIN Ciudad as Ciu1 ON(V2.cod\_ciudad\_destino=Ciu2.codCiudad)

WHERE (Ciu1.nombre=“Jujuy”)

**10. Listar información de aerolíneas que solo tengan vuelos con destino ‘Argentina’. Informar nombre y origen.**

SELECT DISTINCT codAerolinea, nombre, origen

FROM Aeorlinea as A

INNER JOIN Vuelo V ON (A.codAerolinea=V.codAerolinea)

INNER JOIN Ciudad as Ciu ON(V.cod\_ciudad\_destino=Ciu.codCiudad)

WHERE (Ciu.pais=Argentina)

**Ejercicio 3**

**Cine = (#codC, nombreC, direccion)**

**Sala = (#codS, nombreS, descripción, capacidad, #codC)**

**Película = (#codP, nombre, descripción, genero)**

**Funcion = (#codF, #codS, #codP, fecha, hora, ocupación) //ocupación indica # de espectadores de la función**

**1. Reportar información de películas exhibidas en cines de ‘Avellanada’ y que posean funciones en cines de ‘La Plata’.**

SELECT Distinct nombre, p.descripción, genero

FROM Película as P1

INNER JOIN Función as F1 ON (P1.codP=F1.codP)

INNER JOIN Sala as S1 ON (F1.codS=S.codS)

INNER JOIN Cine as C1 ON (S1.codC=C1.codC)

WHERE(C1.direccion LIKE “%Avellaneda%”)

INTERSECT

SELECT Distinct nombre, p2.descripción, genero

FROM Película as P2

INNER JOIN Función as F2 ON (P2.codP=F2.codP)

INNER JOIN Sala as S2 ON (F2.codS=S2.codS)

INNER JOIN Cine as C2 ON (S2.codC=C2.codC)

WHERE(C2.direccion LIKE “%La Plata%”)

**2. Reportar todas las películas que fueron exhibidas en funciones con menos de 30 espectadores. Ordenar por nombre de película**

SELECT Dinstinct nombre, p.descripción, genero

FROM Película as P

INNER JOIN Función as F ON (P.codP=F.codP)

WHERE (ocupación<30)

ORDER BY nombre

**3. Listar nombre y dirección de los cines que exhiban todas las películas.**

SELECT Distinct nombreC, dirección

FROM Cine as C

WHERE Not Exist (

(SELECT codP

FROM Película as P)

EXCEPT

(SELECT codP

FROM Función as F

INNER JOIN Sala as S ON (F.codS=S.codS

WHERE (C.codC=S.codC) )

)

**4. Modificar el nombre a: ‘Sala Darin’, de la sala con código 1000.**

UPDATE Sala

SET nombreS=“Sala Darin”

WHERE codS=1000

**5. Listar nombre y dirección de cines donde se exhiba la película: ‘007 Bond: Sin tiempo para morir’ o que tengan funciones con ocupación durante 2020.**

SELECT nombreC, dirección

FROM Cine as C1

INNER JOIN Sala as S1 ON (C1.codC=S1.codC)

INNER JOIN Función as F1 ON (S1.codS=F1.codS)

INNER JOIN Película as P1 ON (F1.codP=P1.codP)

WHERE (nombre=“007 Bond: Sin tiempo para morir”)

UNION

SELECT Distinct nombreC, dirección

FROM Cine as C2

INNER JOIN Sala as S2 ON (C2.codC=S2.codC)

INNER JOIN Función as F2 ON (S2.codS=F2.codS)

WHERE ( (ocupación>0) and ( YEAR(Fecha)=2020) )

**6. Reportar nombre, descripción y género de películas exhibidas en el Cine: ´Cine X´ pero que no tengan programadas funciones en dicho cine para el dia de hoy.**

SELECT nombre, p.descripción, genero

FROM Película as P

INNER JOIN Función as F ON (P.codP=F.codP)

INNER JOIN Sala as S ON (F.codS=S.codS)

INNER JOIN Cine as C ON (S.codC=C.codC)

WHERE (nombreC=“Cine X”) and Not Exists(

SELECT \*

FROM Función as F2

WHERE Fecha=“19/05/2022” and F.codf=F2.codf

**7. Reportar para cada cine la cantidad de espectadores por película durante 2020. Indicar nombre del cine, nombre de la película y cantidad de espectadores. Ordenar por cine y luego por película.**

SELECT Distinct nombreC, nombre, SUM(ocupación) as cantidad de espectadores

FROM Cine as C

INNER JOIN Sala as S ON (C.codC=S.codC)

INNER JOIN Función as F ON (S.codS=F.codS)

INNER JOIN Película as P ON (F.codP=P.codP)

WHERE( YEAR(fecha)=2020 )

GROUP BY nombreC, nombre

ORDER BY nombreC, nombre

**8. Borrar el cine con nombre ‘Cine China Zorrilla’.**

DELETE FROM Cine

WHERE (nombreC=“Cine China Zorilla”)

**9. Reportar para cada película, nombre y la cantidad de veces que fue exhibida durante 2020.**

SELECT nombre, Count(codF) as Cantidad de veces exhibida

FROM Película as P

INNER JOIN Función as F ON (P.codP=F.codP)

WHERE ( YEAR(fecha)=2020 )

GROUP BY nombre

**10. Listar las películas que no se hayan exhibido en 2020 en ningún cine de La Plata.**

SELECT nombre, p.descripción, genero

FROM Película as P

WHERE Not Exist (

SELECT \*

FROM Función as F

INNER JOIN Sala as S (F.codS=S.codS)

INNER JOIN Cine as C (C.codC=S.codC)

WHERE ( (P.codP=F.codP) and ( YEAR(fecha)=2020) and (dirección LIKE “%La Plata%”))

)

**Ejercicio 4**

**Productos = (idProducto, nombre, presentación, stock, stock mínimo, precioActual)**

**Empleados = (codigoEmp, DNI, nombre,fn, dirección)**

**Clientes = (codigoCte, DNI, nombre, dirección, telefono)**

**Ventas = (codVenta, nroTicket, codigoEmp, codigoCte, fecha, montoTotal)**

**DetalleVentas = (codVenta, idProducto, cantidad, precioUnitario)**

**1. Reportar nombre, dirección y teléfono de clientes que compraron a todos los empleados que viven en su localidad. (Asumir dirección=localidad).**

SELECT C.nombre as NombreCliente, C.dirección as DireccionCliente, teléfono

FROM Cliente as C

WHERE Not Exist(

(SELECT codigoEmp

FROM Empleados

WHERE (C.direccion=E.direccion)

) EXCEPT (

SELECT códigoEmp

FROM Ventas as V

WHERE (C.codigoCte=V.codigoCte)

)

)

**2. Listar para cada empleado, la cantidad de ventas realizadas durante 2019. Reportar DNI, nombre, fn, dirección y cantidad de ventas. El listado debe estar ordenado por nombre y fn.**

SELECT DNI, nombre,fn, dirección, Count(nroTicket) as Cantidad\_de\_Ventas

FROM Empleados as E

INNER JOIN Ventas as V ON (E.codigoEmp=V.codEmp)

WHERE ( YEAR(fecha)=2019 )

GROUP BY DNI, nombre,fn, dirección

ORDER BY nombre, fn

**3. Listar datos personales de los empleados que tengan ventas con más de 50 artículos diferentes.**

SELECT DNI, nombre,fn, dirección

FROM Empleados as E

WHERE 50 < (

SELECT Count(idProducto)

FROM Productos as P

INNER JOIN DetalleVentas as DV ON (P.idProducto=DV.idProducto)

INNER JOIN Ventas as V ON (DV.codVenta=V.codVenta)

WHERE (E.codigoEmp=V.codEmp)

GROUP BY idProducto)

SELECT DNI, nombre,fn, dirección

FROM Empleados as E

INNER JOIN DetalleVentas as DV ON (P.idProducto=DV.idProducto)

INNER JOIN Ventas as V ON (DV.codVenta=V.codVenta)

HAVING COUNT(idproducto) >50

GROUP BY DNI,nombre,fn,dirección,codigoventa

**4. Informar datos personales del mejor cliente. Aquel cuyo monto de ventas realizadas supera al resto de los clientes.**

SELECT DNI, nombre, dirección, teléfono

FROM Cliente as C

INNER JOIN Venta as V (V.codigoCte=C.codigoCte)

Group by DNI, nombre, dirección, teléfono

Having SUM(montoTotal) >= ALL (

SELECT SUM(montoTotal)

FROM Ventas

Gruop by codigoCte

)

**5. Agregar una venta para el empleado Castelli Juan Manuel con nroTicket 1000 con la fecha y monto que desee para el cliente DNI 22369659.**

INSERT INTO Ventas (nroTicket, codigoEmp, codigoCte, fecha, montoTotal)

Values (1000,

SELECT codigoEMP FROM Empleados WHERE (nombre=Castelli Juan Manuel),

SELECT codigoCte FROM Cliente WHERE (DNI=22369659),

19/05/2022,

5)

**6. Listar DNI, nombre, fn y dirección de empleados que realizaron ventas a todos los clientes.**

SELECT DNI, E.nombre, fn, E.direccion

FROM Empleados as E

Where Not Exist(

(SELECT codigoCte

FROM Cliente

) EXCEPT (

SELECT codigoCte

FROM Venta as V

WHERE(E.codigoEmp=V.codigoEmp)

)

)

**7. Reportar información de ventas (nroTicket, empleado, cliente, fecha, montoTotal) que tengan monto total superior a 10000 y el cliente no sea de ´Tandil´.**

SELECT nroTicket, e.nombre as NombreEmpleado, c.nombre as NombreCliente ,fecha, montoTotal

FROM Ventas as V

INNER JOIN Cliente as C ON (V.codigoCte=C.codigoCte)

INNER JOIN Empleados as E ON (V.codigoEmp=E.codigoEmp)

WHERE ( (montoTotal>10000) and (C.direccion NOT LIKE “%Tandil%”) )

**8. Listar datos personales de clientes que realizaron compras en 2019 pero no realizaron compras durante 2020.**

SELECT DNI, nombre, dirección, teléfono

FROM Cliente as C

INNER JOIN Ventas as V ON (C.codigoCte=V.codigoCte)

WHERE ( YEAR(fecha)=2019 and C.codigoCte Not In(

SELECT codigoCte

**Productos =** (*idProducto*, nombre, presentación, stock, stock mínimo, precioActual)  
**Empleados =** (*codigoEmp*, DNI, nombre,fn, dirección)  
**Clientes =** (*codigoCte*, DNI, nombre, dirección, telefono)  
**Ventas =** (*codVenta*, nroTicket, codigoEmp, codigoCte, fecha, montoTotal)  
**DetalleVentas =** (*codVenta, idProducto*, cantidad, precioUnitario)

FROM Ventas

WHERE ( YEAR(fecha)=2018)

)

**9. Listar datos personales de empleados que participaron de ventas con algún producto con precioActual superior a 1000.**

SELECT Distinct DNI, E.nombre, fn, dirección

FROM Empleados as E

INNER JOIN Ventas as V ON (E.codigoEmp=V.codigoEmp)

INNER JOIN DetalleVentas as DV ON (DV.codVenta=V.codVenta)

INNER JOIN Productos as P ON (DV.idProducto=P.idProducto)

WHERE(precioActual>1000)

**10. Listar los datos de los productos que no fueron vendidos durante 2020.**

SELECT nombre, presentación, stock, stock mínimo, precioActual

FROM Productos as P

INNER JOIN DetalleVentas as DV ON (DV.idProducto=P.idProducto)

WHERE codVenta Not in(

SELECT codVenta

FROM Ventas

WHERE ( YEAR(fecha)=2020)

)

**Ejercicio 5**

**Persona = (DNI, Apellido, Nombre, Fecha\_Nacimiento, Estado\_Civil, Genero)**

**Alumno = (DNI, Legajo, Año\_Ingreso)**

**Profesor = (DNI, Matricula, Nro\_Expediente)**

**Titulo = (Cod\_Titulo, Nombre, Descripción)**

**Titulo-Profesor = (Cod\_Titulo, DNI, Fecha)**

**Curso = (Cod\_Curso, Nombre, Descripción, Fecha\_Creacion, Cantidad\_Horas)**

**Alumno-Curso = (DNI, Cod\_Curso, Año, Desempeño, Calificación)**

**Profesor-Curso = (DNI, Cod\_Curso, Fecha\_Desde, Fecha\_Hasta)**

**1. Listar nombre, descripción y fecha de creación de los cursos que se encuentra inscripto el alumno con legajo 1089. Ordenar por nombre y fecha de creación descendentemente.**

SELECT C.Nombre, Descripción, Fecha\_Creacion

FROM Curso as C

INNER JOIN Alumno-Curso as AC ON (C.Cod\_Curso=AC.CodCurso)

INNER JOIN Alumno as A ON (AC.DNI=A.DNI)

WHERE (Legajo= “1089”)

Order By C.Nombre, Fecha\_Creacion Desc

SELECT column1, column2, ...  
FROM table\_name  
ORDER BY column1, column2, ... ASC|DESC;

**2. Agregar un profesor con los datos que prefiera y con título ‘Licenciado en Sistemas’.**

INSERT INTO Profesor (42900457,15,223)

INSERT INTO Titulo-Profesor(SELECT Cod\_Titulo FROM Titulo WHERE (Nombre= “Licenciado en Sistemas”), 42900457,20/05/2002)

**3. Listar el DNI, apellido, nombre y matrícula de aquellos profesores que posean menos de 5 títulos. Dicho listado deberá estar ordenado por apellido y nombre.**

SELECT P.DNI, Apellido, P.nombre, matricula

From Persona as P

INNER JOIN Profesor as Prof ON (P.DNI=Prof.DNI)

WHERE 5>(SELECT Count(\*)

FROM Titulo-Profesor as TP

WHERE (TP.DNI=P.DNI)

GROUP BY Cod.Titulo)

Order by apellido, nombre

**4. Listar el DNI, apellido, nombre y cantidad de horas que dicta cada profesor.**

SELECT P.DNI, Apellido, P.nombre, SUM(Cantidad\_Horas) as Cantidad\_HorasTotal

From Persona as P

INNER JOIN Profesor as Prof ON (P.DNI=Prof.DNI)

LEFT JOIN Profesor-Curso as PC ON (Prof.DNI=PC.DNI)

INNER JOIN Curso as C ON (PC.Cod\_Curso=C.Cod\_Curso)

GROUP BY P.DNI, Apellido, P.nombre

**5. Listar nombre,descripción del curso que posea más alumnos inscriptos y del que posea menos alumnos inscriptos durante 2016.**

SELECT Nombre, Descripcion

FROM Curso as C

INNER JOIN Alumno-Curso as AC ON (C.Cod\_Curso=AC.Cod\_Curso)

INNER JOIN Alumno as A ON (AC.DNI=A.DNI)

WHERE (AC.AÑO=2016)

Group By Nombre, Descripcion

HAVING Count(A.DNI) >=ALL( SELECT Count(A1.DNI)

FROM Alumno as A1

INNER JOIN Alumno-Curso as AC1 ON (A1.DNI=AC1.DNI)

GROUP BY A3.DNI

)

UNION

SELECT Nombre, Descripcion

FROM Curso as C2

INNER JOIN Alumno-Curso as AC2 ON (C2.Cod\_Curso=AC2.Cod\_Curso)

INNER JOIN Alumno as A2 ON (AC2.DNI=A2.DNI)

WHERE (AC2.AÑO=2016)

Group By Nombre, Descripcion

HAVING Count(A2.DNI) <=ALL( SELECT Count(A3.DNI)

FROM Alumno as A3

INNER JOIN Alumno-Curso as AC3 ON (A3.DNI=AC3.DNI)

GROUP BY A3.DNI

)

**6. Listar el DNI, apellido, nombre, género y fecha de nacimiento de los alumnos inscriptos al curso con nombre “tuning de oracle” en 2019 y que no tengan calificación superior a 5 en ningún curso.**

SELECT Distinct P.DNI, Apellido, P.Nombre, Genero, Fecha\_Nacimiento

FROM Persona as P

INNER JOIN Alumno-Curso as AC ON (P.DNI=AC.DNI)

INNER JOIN Curso as C ON (AC.Cod\_Curso=C.Cod\_Curso)

WHERE ( (C.Nombre=“tuning de oracle”) and (AC.Año=2019) and 5>=ALL(

SELECT Calificacion

FROM Alumno-Curso as AC1

WHERE(AC1.DNI=P.DNI)

)

)

**7. Listar el DNI, Apellido, Nombre, Legajo de alumnos que realizaron cursos durante 2018 pero no cursaron durante 2019.**

SELECT Distinct P.DNI, Apellido, P.Nombre, Legajo

FROM Persona as P

INNER JOIN Alumno as A ON (P.DNI=A.DNI)

INNER JOIN Alumnos-Curso as AC ON (A.DNI=AC.DNI)

WHERE ( (AC.Año=2018) and p.dni not in(SELECT ac2.dni

FROM Alumno-Curso as AC2

WHERE ((AC2.DNI=P.DNI) and (AC2.Año=2019))

)

)

**8. Listar nombre, apellido, DNI, fecha de nacimiento, estado civil y género de profesores que tengan cursos activos actualmente o tengan de alumno al alumno DNI:34567487.**

SELECT Distinct P.Nombre, Apellido, P.DNI, Fecha\_Nacimiento, Estado\_Civil, Genero

FROM Persona as P

INNER JOIN Profesor-Curso as PC ON (P.DNI=PC.DNI)

WHERE ( (YEAR(Fecha\_Hasta)>2022) and (MONTH(Fecha\_Hasta)>05) and (DAY(Fecha\_Hasta) > 20) ) or cod\_curso in ( select cod\_curso from alumno\_curso ac where ac.dni=34567487)

UNION

SELECT Distinct P2.Nombre, Apellido, P2.DNI, Fecha\_Nacimiento, Estado\_Civil, Genero

FROM Persona as P2

INNER JOIN Profesor-Curso as PC2 ON (P2.DNI=PC2.DNI)

INNER JOIN Curso as C ON (PC2.Cod\_Curso=Curso.CodCurso)

INNER JOIN Alumno-Curso as AC ON (C.Cod\_Curso=AC.Cod\_Curso)

WHERE (AC.DNI=34567487)

**9. Dar de baja el alumno con DNI 38746662. Realizar todas las bajas necesarias para no dejar el conjunto de relaciones en estado inconsistente.**

DELETE FROM Alumno

WHERE(DNI=38746662)

DELETE FROM Alumno-Curso

WHERE(DNI=38746662)

DELETE FROM Persona

WHERE(DNI=38746662)

**10. Listar para cada curso nombre y la cantidad de alumnos inscriptos en 2020**

SELECT C.nombre, Count(DNI)

FROM Curso as C

INNER JOIN Alumno-Curso as AC ON (C.Cod\_Curso=AC.Cod\_Curso)

WHERE (Año=2020)

Group by Cod\_Curso

**Ejercicio 6  
Departamento = (*codDepto*, nombreD, funciones, fecha\_creacion)** // funciones es una descripción de las funciones más relevantes  
**Area = (*codArea*, codDepto, nombreA, descripción)** // codDepto es el departamento al cual corresponde esa área  
**Proyecto = (*codProyecto*, nombreProyecto, avance, descProyecto, objetivos**) // descripción del proyecto  
**Aplicacion = (*codAplicacion,*codProyecto, codArea*,* fechaIni, porcentajeFinalizado, fechaFin,  
fechaPrevistaFin)** // descripción de aplicaciones del proyecto en determinadas áreas indicando evolución del mismo

**1. Listar aplicaciones de proyectos que se terminaron luego de la fecha prevista de fin. Indicar nombre del proyecto, área de aplicación, departamento correspondiente, fecha de inicio y fin, y fecha prevista de fin. Ordenar por nombre de proyecto y área.**

SELECT distinct nombreProyecto, nombreA, nombreD, fechaIni, fechaFin, fechaPrevistaFin

FROM Aplicación as Ap

INNER JOIN Proyecto as P ON (Ap.codProyecto=P.codProyecto)

INNER JOIN Area ON (Ap.codArea=Area.codArea)

INNER JOIN Departamento as D ON (Area.codDepto=D.codDepto)

WHERE (fechaFin> Fechaprevistafin)

Order by nombreProyecto, nombreA

**2. Reportar para cada área la cantidad de proyectos finalizados durante 2020. Informar nombre de área y cantidad de proyectos finalizados.**

SELECT Distinct nombreA, Count(codAplicacion)

FROM Area as Ar

INNER JOIN Aplicacion as Ap ON (Ar.codArea=Ap.codArea)

WHERE ( YEAR(fechaFin)=2020) )

Group by codArea, nombreA **3. Listar información de departamentos que no tengan proyectos terminados.**

**Select nombreD, funciones, fecha\_creacion**

**From departamento d**

**Inner join área A on (a.codDepto=d.codDepto)**

**Inner join Aplicacion A1 on (a.codArea=a1.codArea)**

**Where not exist (select codproyecto**

**From proyecto p**

**Where (p.avance=100) and (p.codproyecto=a1.codproyecto)**

SELECT nombreD, funciones, fecha\_creacion

FROM Departamento as D

INNER JOIN Area as Ar ON (D.codDepto=Ar.codDepto)

INNER JOIN Aplicación as AP ON (AP.codArea=Ar.codArea)

WHERE Not Exist(SELECT \*

FROM Proyecto as P

WHERE (P.avanceFinalizado=100) and (P.codProyecto=AP.codProyecto)

) **4. Listar información de proyectos que tengan no menos 50 aplicaciones finalizadas. Ordenar por cantidad de aplicaciones.**

Select nombreProyecto, avance, descProyecto, objetivos, count (codaplicacion) as aplicacionestotal

From Proyecto p

Inner join aplicacion a on (a.codproyecto= p.codproyecto)

Group by codproyecto, nombreProyecto, avance, descProyecto, objetivos

Having 50 <= Count(codaplicacion)

Order by count(codaplicacion)

SELECT nombreProyecto, avance, descProyecto, objetivos, Count(codAplicacion) as Cantidad\_Aplicaciones

FROM Proyecto as P

INNER JOIN Aplicación as AP ON(AP.codProyecto=P.proyecto)

WHERE (porcentajeFinalizado=100)

Group by codProyecto, nombreProyecto, avance, descProyecto, objetivos

Having Count(codAplicacion) >= 50

Order by Cantidad\_Aplicaciones **5. Agregar un nuevo proyecto con la información que desee.**

INSERT INTO Proyecto (nombreProyecto, avance, descProyecto, objetivos)

VALUES (69, Proyecto X, 34, secreto, tambiensecreto ) **6. Listar información de proyectos que tengan aplicación en todas las áreas.**

**Select nombreProyecto, avance, descProyecto, objetivos**

**From proyecto p**

**Where not exist ((select codarea**

**From area)**

**Except**

**(select codarea**

**From aplicacion a**

**Where (a.codproyecto=p.codproyecto)**

**)**

**)**

SELECT nombreProyecto, avance, descProyecto, objetivos

FROM Proyecto as P

WHERE NOT EXIST (

(

SELECT codArea

FROM Area

) EXCEPT (

SELECT codArea

From Area

INNER JOIN Aplicación as A

WHERE P.codProyecto = A.codProyecto

)

)

**7. Listar información de departamentos que no tengan aplicaciones registradas.**

SELECT nombreD, funciones, fecha\_creacion

**From departamento d**

**Inner join area a on (a.coddepto=d.coddepto**

**Where not exist ( select \***

**From aplicación a1**

**Where a1.codarea=a.codarea)**

SELECT nombreD, funciones, fecha\_creacion

FROM Departamento as D

INNER JOIN Area as A ON (D.codDepto=A.codDepto)

WHERE Not Exist (

SELECT \*

FROM Aplicación

WHERE (A.codArea=Aplicación.codArea)

) **8. Listar información de proyectos que tenga alguna aplicación con fecha de comienzo inferior a 2010 y aún no se terminó. Informar nombre de proyecto, área, fecha inicio y fecha prevista fin.**

**Select distinct** nombreProyecto, nombrea,fechaini,fechaprevistafin

From proyecto p

Inner join aplicación a on (a.codproyecto=p.codproyecto)

Inner join area a1 on (a1.codarea=a.codarea)

Where Year(fechaini)<2010 and (100>porcentajefinalizado)

SELECT Distinct nombreProyecto, nombreA, fechaIni, fechaPrevistaFin

From Proyecto as P

INNER JOIN Aplicación as Ap ON (P.codProyecto=Ap.codProyecto)

INNER JOIN Area as Ar ON (Ap.codArea=Ar.codArea)

WHERE ( YEAR(fechaIni)<2010 and (porcentajeFinalizado<>100) )

**9. Listar información de de todas las áreas que componen el departamento de ‘Redes’.**

**Select distinct** codDepto, nombreA, descripción

From área a

Inner join departamento d on (d.codarea=a.codarea)

Where (nombred=”redes”)

SELECT Distinct codDepto, nombreA, descripción

FROM Area as A

INNER JOIN Departamento as D ON (A.codDepto=D.codDepto)

WHERE (nombreD = “Redes”) **10. Listar información de proyectos que sólo tengan aplicaciones finalizadas durante 2019.**

**Select distinct** nombreProyecto, avance, descProyecto, objetivos

From proyecto p

Inner JOIN Aplicación as Ap ON (P.codProyecto=Ap.codProyecto)

Where YEAR(aP.fechafin)=2019 and ap.codaplicacion not in (select codaplicacion

From aplicación a

Where (a.fechafin !=2019)

)

SELECT nombreProyecto, avance, descProyecto, objetivos

FROM Proyecto as P

INNER JOIN Aplicación as Ap ON (P.codProyecto=Ap.codProyecto)

WHERE ( YEAR(Ap.fechaFin)=2019) and not exist(

SELECT \*

FROM Aplicación as Ap1

WHERE ( YEAR(Ap1.fechaFin)<>2019) and (Ap1.codProyecto=P.codProyecto) )

)

**PARCIAL 2014**

**3. SQL Y AR**

1) SELECT Mar.#marca, Mar.nombre, Mar.descripcion, Mod.#modelo, Mod.nombre, Mod.descripcion

FROM Marca as Mar

INNER JOIN Vehículo as V ON (Mar.#marca=V.#marca)

INNER JOIN Modelo as Mod ON (V.#modelo=Mod.#modelo)

Group by Mar.#marca, Mar.nombre, Mar.descripcion, Mod.#modelo, Mod.nombre, Mod.descripcion

HAVING Count(#motor) > 100

2) SELECT #concesionario, calle, nro, ciudad, teléfono, razón social

FROM Concesionario as C

WHERE not exist (

(Select #marca

FROM Marca

) Except( Select #marca

FROM Vehiculo as V

WHERE (V.#concesionario=C.#concesionario)

)

Order by razón social

3) SELECT Mod.nombre

FROM Modelo as Mod

INNER JOIN Vehiculo as V ON (Mod.#Modelo=V.#Modelo)

INNER JOIN Concesionario as C ON (V.#concesionario=C.#concesionario)

WHERE (C.ciudad = “La Plata”) and not exist (

SELECT \*

FROM Concesionario as C2

WHERE (C2.ciudad= “Trelew”) and (C.#concesionario=C2.#concesionario)

)

Select m.nombre

From modelo m

Inner join vehiculo v on (v.#modelo=m.#modelo)

Inner join concesionario c on (c.#concesionario=v.#concesionario)

Where ciudad=”La plata” and AND M.#modelo NOT IN (

SELECT #modelo

FROM Vehiculo V2

INNER JOIN Concesionario C2 ON V2.#concesionario = C2.#concesionario

WHERE C2.ciudad = 'Trelew'

)

4)

SELECT calle, nro, ciudad, telefono, razón social

FROM Concecionario as C

INNER JOIN Vehiculo as V ON (C.#concesionario=V.#concesionario)

INNER JOIN Modelo as Mod ON (V.#Modelo=Mod.#Modelo)

INNER JOIN Marca as Mar ON (V.#Marca= Mar.#Marca)

WHERE (Mar.nombre= “Ford”) and (Mod.nombre LIKE “Fo%”)

Select calle, nro, ciudad, telefono, razón social

From concesionario c

Inner join vehiculo v on (v.#concesionario=c.#concesionario)

Inner join modelo m on (m.#modelo=v.#modelo)

Inner join marca ma on (m.#marca=v.#marca)

Where Ma.nombre=“Ford” and m.nombre like “Fo%”

5)

-

6)

6)

*7)(*

7)

**Ejercicio 7**

**Cartero = (DNI,nombreYApe,** **dirección, teléfono)**

**Sucursal = (IDSUC, nombreS,direcciónS, teléfonoS)**

**Envio = (NROENVIO, DNI, IDCLIENTEEnvia, IDCLIENTERecibe, IDSUC, fecha, recibido, fechaRecibe, direcciónEntrega) //recibido es blanco sino se entregó aún el envío**

**Cliente = (IDCLIENTE, DNI,nombreYApe, dirección, teléfono)**

1. **Reportar nombre y apellido, dirección y teléfono del cartero que realizó más envíos.**

Select DNI, nombreyape, dirección, teléfono

From cartero c

Inner join Envio e on (c.dni=e.dni)

Group by c.dni, nombreyape, dirección, teléfono

Having count (nroenvio) >=all (select count(e1.nroenvio)

From envio e1

Group by DNI)

Select nombreYApe, dirección, teléfono

From Cartero

Natural Join Envio

Group by DNI,nombreYApe, dirección, teléfono

Having Count(NROENVIO) >=all (Select Count(E1.NROENVIO)

From Envio as E1

Group by E1.dni

)

1. **Listar para cada sucursal, la cantidad de envíos realizados a alguna dirección de envío que contenga el string ‘Ju’. Informar nombre de sucursal y cantidad de envíos correspondiente. Ordenar por nombre sucursal.**

Select nombreS, count(nroenvio)

From sucursal s

Inner join envio e on (e.idsuc=s.idcsuc)

Where direccionentrega like “%Ju%”

Group by idsuc, nombreS

Order by nombreS

Select nombreS, Count(NROENVIO) as Cantidad de envois

From Sucursal

Natural Join Envio

Where ( direccionEntrega LIKE “%Ju%”)

Group by IDSUC, nombresS

Order by nombreS

1. **Listar datos personales de carteros que entregaron envíos a todas las sucursales.**

Select DNI,nombreYApe, dirección, teléfono

From cartero c

Where not exist ( (select idsuc

From sucursal)

Except (select idsuc from entrega e where e.DNI=c.DNI))

Select DNI, nombreYApe, dirección, teléfono

From Cartero as C

Where not exist ( (Select IDSUC

From Sucursal

) Except ( Select IDSUC

From Envio as E

Where (E.DNI=C.DNI)

)

)

1. **Informar cantidad de envíos no entregados del mes de mayo de 2019.**

Select Count (nroenvio)

From envio e

Where Year(fecha)=2019 and month(fecha)=05 and recibido=” ”

Select NROENVIO, Count() as CantNoEntregados

From Envio as E

Where ( (recibido= “ “) and ( YEAR(fecha)=2019) and ( Month(fecha)=Mayo) )

1. **Borrar al cliente con IDCLIENTE: 334.**

Delete from cliente

Where idcliente=334

Delete from envio

Where idclienterecibe=334

Delete from envio

Where idclienteenvia=334

Delete From Cliente

Where (IDCLIENTE = 334)

Delele from Envio

Where (IDCLIENTEEnvia = 334)

Delele from Envio

Where (IDCLIENTERecibe = 334)

1. **Listar datos personales de carteros que no entregaron envío a clientes(receptor) residentes en ‘La Plata’ pero si realizaron envíos de clientes (emisor) que residen en ‘Wilde’ (el cliente que envía vive en Wilde).**

intersección(

Select distinct DNI,nombreYApe, dirección, teléfono

From cartero c

Inner join envio e on (e.dni=c.dni)

Inner join cliente c1 on (c1.idcliente=e.clienteenvia)

Where c1.direccion like “%wilde%” and c.dni not in (select dni from envio where direccionentrega like “%La Plata%”)

Select Distinct C.DNI, C.nombreYApe, C.dirección, C.telefono

From Cartero as C

Natural Join Envio as E

INNER JOIN Cliente as Clt ON (E.IDCLIENTEEnvia=Clt.IDCLIENTE)

Where( ( Clt.direccion LIKE “%Wilde%”) and DNI not in ( Select E1.DNI

From Envio as E1

INNER JOIN Cliente as Clt1 ON (E1.IDCLIENTERecibe=Clt1.IDCLIENTE)

Where( (Clt.direccion LIKE “%La Plata%”) and E1.DNI=E=DNI)

)

)

1. **Reportar información de sucursales que realizaron envíos durante 2020 o que tengan dirección en Tucuman.**

Select Distinct IDSUC, nombreS, direcciónS, teléfonos

From sucursal s

Left join envio e on (e.idsuc=s.idsuc)

Where direccionS like “%Tucuman%” or YEAR(e.fecha)=2020

Select Distinct IDSUC, nombreS, direcciónS, teléfonos

From Sucursal as S

Left join Envio as E ON (S.IDSUC = E.IDSUC)

Where ( (YEAR(fecha)=2020) OR (direccionS = “Tucuman”) )

1. **Listar datos personales de clientes que no realizaron envíos a la sucursal con nombre ‘La Amistad 1’.**

Select DNI,nombreYApe, dirección, teléfono

From Cliente c

Where idcliente not in (select idclienteenvia

From envio e

Inner join sucursal s on (s.idsuc=e.idsuc)

Where s.nombres=“La Amistad 1”)

Select IDCLIENTE, Cte.DNI, Cte.nombreYApe, Cte.dirección, Cte.teléfono

From Cliente as Cte

Where IDCLIENTE not in ( (Select IDCLIENTEEnvia

From Envio

Natural Join Sucursal

Where (nombreS = “La Amistad 1”)

)

1. **Listar datos personales de carteros que realizaron envíos durante 2019 a clientes con DNI inferior a 27329882.**

select DNI,nombreYApe, dirección, teléfono

from cartero c

where DNI in (select e.dni from envio e

inner join cliente c1 on (c1.idcliente=e.idclienterecibe)

where year(fecha)=2019 and c1.dni <=27329882)

Select Distinct C.DNI, C.nombreYApe, C.dirección, C.teléfono

From Carterto as C

Natural Join Envio as E

INNER JOIN Cliente as Cte ON (E.IDCLIENTEEnvia=Cte.IDCLIENTE)

Where (( Year(fecha)=2019) and Cte.DNI<27329882)

**10. Listar los datos de los carteros que aún no hayan realizado ninguna entrega**

Select Distinct C.DNI, C.nombreYApe, C.dirección, C.teléfono

From Carterto as C

WHERE DNI NOT IN ( SELECT DNI

FROM envio

Where recibió is not null

)

Select DNI,nombreYApe, dirección, teléfono

From cartero c

Left join envio e on (e.dni=c.dni)

Group by DNI,nombreYApe, dirección, teléfono

Having count(nroentrega)=0