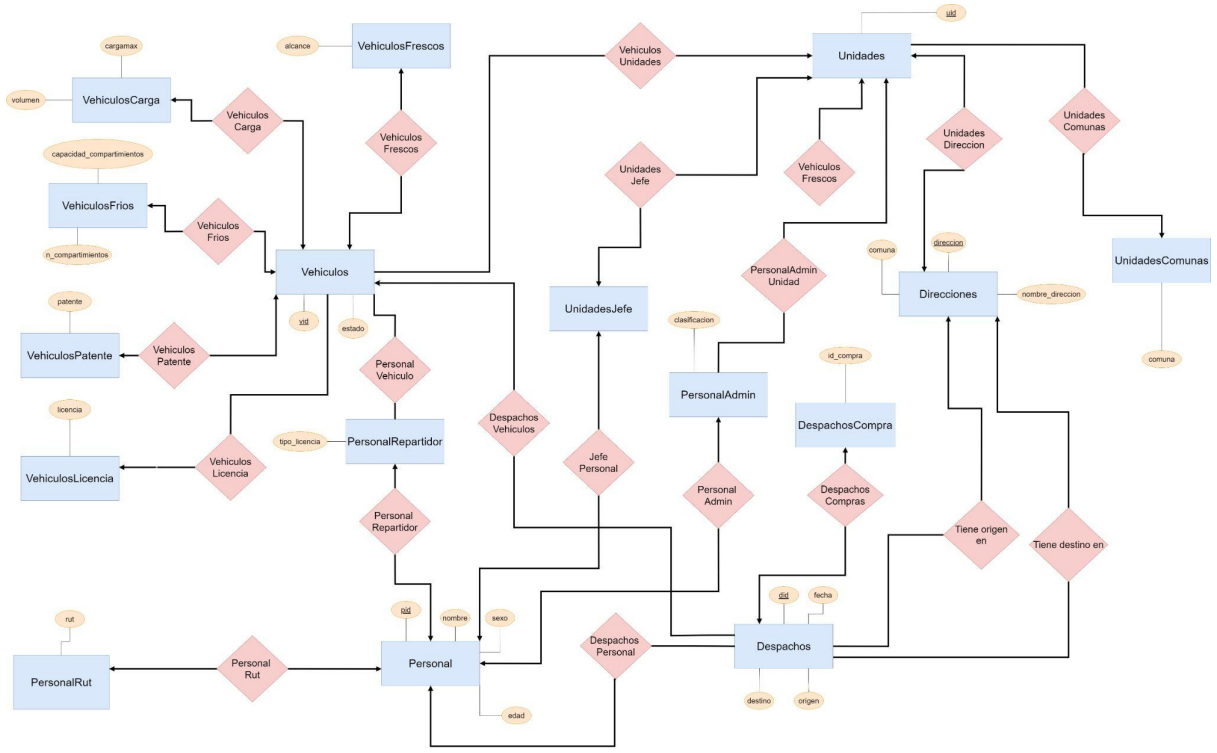
Entrega 2

Diagrama E/R



Esquema Relacional

<u>Unidades</u>

Unidades{

- uid INT PRIMARY KEY
- direccion INT
- FOREIGN KEY(direccion) REFERENCES Direcciones(direccion)

}

UnidadesJefe{

- uid INT PRIMARY KEY
- pid INT UNIQUE
- FOREIGN KEY(pid) REFERENCES Personal(pid)
- FOREIGN KEY (uid) REFERENCES Unidades(uid)

}

UnidadesComunas{

- uid INT
- comuna VARCHAR(100)
- FOREIGN KEY (uid) REFERENCES Unidades(uid)

}

Vehiculos

Vehiculos{
- vid INT PRIMARY KEY
- estado VARCHAR(100)
- uid INT
- FOREIGN KEY(uid) REFERENCES Unidades(uid)

}

VehiculosLicencia{
- vid INT PRIMARY KEY
- licencia VARCHAR(100))
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

VehiculosPatente{
- vid INT PRIMARY KEY
- patente VARCHAR(100)
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

VehiculosFrescos{
- vid INT PRIMARY KEY
- alcance FLOAT
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

VehiculosFrios{
- vid INT PRIMARY KEY
- n_compartimientos INT
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

VehiculosCarga{
- vid INT PRIMARY KEY
- volumen INT
- cargamax INT
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

Despachos

Despachos{
- did INT PRIMARY KEY
- pid INT
- fecha DATE
- origen VARCHAR(100)
- destino VARCHAR(100)

- vid INT
- FOREIGN KEY (pid) REFERENCES Personal(pid)
- FOREIGN KEY (origen) REFERENCES Direcciones(direccion)
- FOREIGN KEY (destino) REFERENCES Direcciones(direccion)
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

DespachosCompra{

- did INT PRIMARY KEY
- id_compra INT
- FOREIGN KEY (did) REFERENCES Despachos(did)

}

Direcciones

Direcciones{

- direccion INT PRIMARY KEY
- nombre_direccion VARCHAR(100)
- comuna VARCHAR(100)

}

Personal

Personal{

- pid INT PRIMARY KEY
- nombre VARCHAR(100)
- sexo VARCHAR(100)
- edad INT

}

PersonalAdmin{

- pid INT PRIMARY KEY
- clasificacion VARCHAR(100)
- uid INT
- FOREIGN KEY (pid) REFERENCES Personal(pid)
- FOREIGN KEY (uid) REFERENCES Unidades(uid)

}

PersonalRepartidor{

- pid INT PRIMARY KEY
- tipo_licencia VARCHAR(100)
- vid INT
- FOREIGN KEY (pid) REFERENCES Personal(pid)
- FOREIGN KEY (vid) REFERENCES Vehiculos(vid)

}

PersonalRut{

- pid INT PRIMARY KEY

- rut VARCHAR(100)
- FOREIGN KEY (pid) REFERENCES Personal(pid)

}

Justificación BCNF

En la tabla Unidades del archivo unidades.csv se nos dieron los siguientes atributos id, direccion, jefe y comuna_cobertura. Decidimos cambiar id por uid y comuna_cobertura por comuna para simplificar la busqueda de datos dentro de nuestra base de datos. Luego nos dimos cuenta que:

- uid -> direccion
- uid -> Jefe
- uid -> comuna

Por ello, para preservar BCNF creamos Unidades(uid, direccion), UnidadesJefe(uid, jefe) y por ultimo UnidadesComunas(uid, comuna)

En el archivo Vehiculos.csv se nos dieron los siguientes atributos: id, patente, estado, tipo, volumen(m3), carga_maxima(ton), alcance(km), cantidad_compartimentos, capacidad_compartimentos(kg), unidad. Decidimos cambiar id por vid, volumen(m3) por volumen, carga_maxima(ton) por cargamax, alcance(km) por alcance, cantidad_compartimientos por n_compartimientos y unidad por uid para simplificar la busqueda de datos dentro de nuestra base de datos. Luego nos dimos cuenta que:

- vid -> patente
- vid -> licencia
- vid, patente, licencia -> estado, tipo, volumen, cargamax, alcance, n_compartimientos, capacidad_compartimientos, unidad

Por ello para preservar BCNF creamos la tabla Vehiculos(vid, estado, uid), VehiculosPatente(vid, patente) y VehiculosLicencia(vid, licencia). El resto de las tablas fueron creadas dependiendo de la categoría del vehículo donde tenemos VehiculosCarga(vid, volumen, cargamax) para tener la informacion de los vehículos de carga, VehiculosFrescos(vid, alcance) para la informacion de los vehiculos para productos frescos y VehiculosFrios(vid, n_compartimientos, capacidad_compartimientos) para vehículos de cadena fría.

En el archivo despachosV2.csv se nos dieron los siguientes atributos: id, fecha, dirección_origen, dirección_destino, vehiculo, repartidor. Decidimos cambiar dirección_origen y dirección_destino por simplemente origen y destino para acortar la busquéda. También cambiamos vehiculo y repartidor por vid y pid para así poder relacionar de manera más sencilla los valores con sus respectivas tablas referenciadas. También cambiamos id por did para diferenciarlo de otros id de otras tablas. Nos dimos cuenta que:

- did -> id_compra

Por ello para preservar BCNF creamos la tabla Despachos(did, fecha, origen, destino, vid, pid) y DespachosCompra(did, id_compra).

En el archivo personalV2.csv se nos dieron los siguientes atributos: id, nombre, rut, sexo, edad, clasificación, unidad, tipo_licencia, vehículo. Decidimos cambiar unidad y vehículo por uid y vid respectivamente debido a que así es más sencillo relacionar los valores con sus respectivas tablas referenciadas. También cambiamos id por pid para así diferenciar los ids de las otras tablas. Nos dimos cuenta que:

- pid -> rut
- pid -> nombre, sexo, edad
- pid -> clasificacion, uid
- pid -> tipo_licencia, vid
- did -> id_compra

Por ello para preservar BCNF creamos la tabla Personal(pid,nombre,sexo,edad), PersonalAdmin(pid,clasificacion,uid), PersonalRepartidor(pid,tipo_licencia,vid), PersonalRut(pid,rut)

Consultas SQL

1. Muestre las direcciones de todas las unidades de la empresa de despachos

```
SELECT Unidades.uid, Direcciones.nombre_direccion,
Direcciones.comuna
FROM Direcciones, Unidades
WHERE Unidades.direccion = Direcciones.direccion
ORDER BY Unidades.uid;
```

2. Ingrese una comuna. Muestre todos los jefes de tiendas ubicadas en dicha comuna.

```
SELECT Despachos.did, VehiculosPatente.patente
FROM Despachos, Vehiculos, Direcciones, VehiculosPatente
WHERE Direcciones.comuna LIKE '%$comuna%' AND EXTRACT(YEAR FROM
Despachos.fecha) = '$ano' AND Despachos.vid = Vehiculos.vid AND
Direcciones.direccion = Despachos.destino AND Vehiculos.vid =
VehiculosPatente.vid
ORDER BY Despachos.did;
```

3. Seleccione un tipo de producto. Muestre todas las tiendas que venden al menos un producto de dicha categoría.

```
SELECT DISTINCT Vehiculos.vid, VehiculosPatente.Patente
FROM UnidadesComunas, Unidades, Vehiculos, VehiculosPatente
WHERE UnidadesComunas.comuna LIKE '%$comuna%' AND
UnidadesComunas.uid = Unidades.uid AND Unidades.uid =
Vehiculos.uid AND Vehiculos.vid = VehiculosPatente.vid
ORDER BY Vehiculos.vid;
```

4. Ingrese un tipo de vehículo y seleccione dos números. Muestre todos los despachos realizados por un vehículo del tipo ingresado, y cuyo repartidor tiene una edad entre el rango seleccionado.

```
SELECT Despachos.did, Personal.nombre
FROM Despachos, Personal, PersonalRepartidor, Vehiculos,
VehiculosLicencia
WHERE VehiculosLicencia.licencia LIKE '%$licencia%' AND
Personal.edad BETWEEN $desde AND $hasta AND Despachos.vid =
Vehiculos.vid AND Despachos.pid = PersonalRepartidor.pid AND
Personal.pid = PersonalRepartidor.pid AND PersonalRepartidor.vid =
Vehiculos.vid AND Vehiculos.vid = VehiculosLicencia.vid
```

5. Ingrese dos comunas. Encuentre los jefes de las unidades que realizan despachos a ambas comunas.

```
SELECT Unidades.uid, Personal.nombre
FROM Personal, Unidades, UnidadesJefe, (SELECT com2.uid AS cuid2,
com2.comuna AS comcomuna2
                                        FROM UnidadesComunas AS
com2
                                        WHERE com2.comuna LIKE
'%$comuna2%') AS Cta2, (SELECT com.uid AS cuid, com.comuna AS
comcomuna

FROM UnidadesComunas AS com

WHERE com.comuna LIKE '%$comuna1%'

) AS Cta1
WHERE Cta1.cuid = Cta2.cuid2 AND UnidadesJefe.uid = Unidades.uid
AND UnidadesJefe.uid = Cta1.cuid AND UnidadesJefe.pid =
Personal.pid
```

6. Ingrese un tipo de vehículo. Encuentre la unidad que maneja más vehículos de ese tipo.

```
SELECT DISTINCT Unidades.uid
FROM Vehiculos, VehiculosLicencia, Unidades, (SELECT Uni.uid AS id,
COUNT(*) AS cty
                                        FROM
VehiculosLicencia AS VehL, Unidades AS Uni, Vehiculos AS Veh
                                        WHERE VehL.licencia
LIKE '%$vehiculo
%' AND Veh.uid = Uni.uid AND Veh.vid = VehL.vid
                                        GROUP BY Uni.uid) AS
Cuenta
WHERE VehiculosLicencia.licencia LIKE '%$vehiculo
%' AND  Cuenta.cty IN (SELECT MAX(cty2)

FROM (SELECT COUNT(*) AS cty2
```

```
FROM VehiculosLicencia AS VehL2, Unidades AS Uni2, Vehiculos AS
Veh2

WHERE VehL2.licencia LIKE '%$vehiculo
%' AND Veh2.uid = Uni2.uid AND Veh2.vid = VehL2.vid

GROUP BY Uni2.uid) AS Uwu)
    AND Vehiculos.uid = Unidades.uid
    AND Cuenta.id = Unidades.uid
    AND VehiculosLicencia.vid = Vehiculos.vid
```

Supuestos

En el archivo personal.csv habían personas con mismo nombre e id que tenían distintas edades por lo que eliminamos la información de las edades distintas quedándonos con la edad real (dada en los issues de Github) sin eliminar los datos extras que habían en las columnas erróneas.