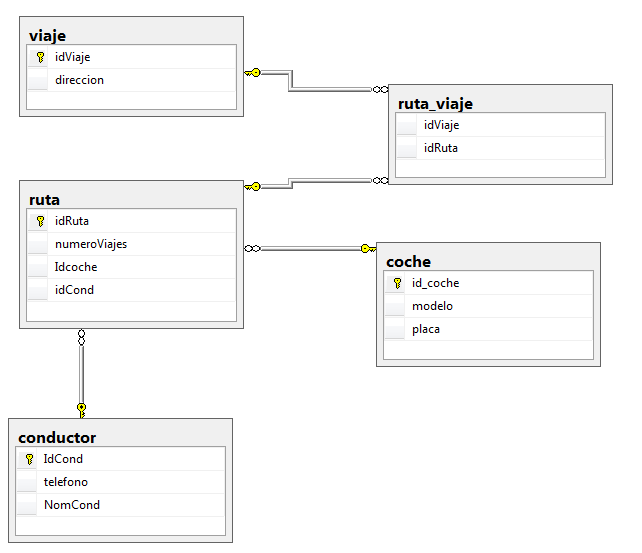
**Ejercicio 4 – SQL (4.4 puntos)**

A continuación se describen las características principales del sistema de bases de datos de UBER.

La siguiente figura muestra el modelo conceptual del sistema de base de datos.

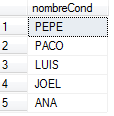


Los archivos **BD-CreacionTablas.txt** y **BD-InsercionTuplas.txt** contienen las instrucciones SQL necesarias para definir la estructura de la base de datos del sistema de UBER y para insertar las tuplas que determinan el estado inicial de la base de datos.

Ingresa al DBMS SQLServer, crea una nueva base de datos llamada **UBER**, ejecuta las sentencias SQL incluidas en los dos archivos mencionados, y posteriormente responde las siguientes preguntas elaborando las consultas que correspondan. Después de resolver cada consulta, copia y pega la sentencia *select* y la tabla de resultados después de cada pregunta.

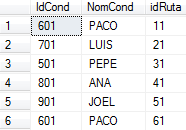
1. **[0.5] Mostrar los nombres (ordenados Z .. A) de los conductores sin repeticiones y la columna se debe llamar nombreCond.**

SELECT DISTINCT c.NomCond AS 'nombreCond' FROM conductor c ORDER BY c.NomCond DESC



1. **[0.5] Obtener el id del conductor, su nombre y los id de la rutas en las que trabaja.**

SELECT c.IdCond, c.NomCond, r.idRuta FROM conductor c INNER JOIN ruta r ON c.IdCond = r.idCond



1. **[0.5] Listar el (o los) id del conductor (es) que trabaja (n) en perisur.**

SELECT c.IdCond FROM conductor c INNER JOIN ruta r ON c.IdCond = r.idCond INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta INNER JOIN viaje v ON rv.idViaje = v.idViaje WHERE v.direccion = 'perisur'



1. **[0.6] Los Ids de las rutas con el mayor número de viajes.**

SELECT TOP(3) r.idRuta FROM ruta r ORDER BY r.numeroViajes DESC

*(Donde el top “3” se puede cambiar al número de rutas necesitadas, pues no se especifica cuantas rutas deben de ser escogidas)*



1. **[0.5] Obtener el promedio de viajes del conductor PEPE.**

SELECT AVG(r.numeroViajes) FROM ruta r INNER JOIN conductor c ON r.idCond = c.IdCond WHERE c.NomCond = 'PEPE'



1. **[0.6] Listar el modelo de los autos que tengan viajes en chapultepec o reforma, pero no en lomas.**

SELECT co.modelo FROM conductor c

INNER JOIN ruta r ON c.IdCond = r.idCond

INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta

INNER JOIN viaje v ON rv.idViaje = v.idViaje

INNER JOIN coche co ON r.Idcoche = co.id\_coche

WHERE v.direccion = 'chapultepec' AND v.direccion != 'lomas' UNION (SELECT co.modelo FROM conductor c

INNER JOIN ruta r ON c.IdCond = r.idCond

INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta

INNER JOIN viaje v ON rv.idViaje = v.idViaje

INNER JOIN coche co ON r.Idcoche = co.id\_coche

WHERE v.direccion = 'reforma')

EXCEPT (SELECT co.modelo FROM conductor c

INNER JOIN ruta r ON c.IdCond = r.idCond

INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta

INNER JOIN viaje v ON rv.idViaje = v.idViaje

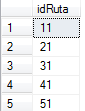
INNER JOIN coche co ON r.Idcoche = co.id\_coche

WHERE v.direccion = 'lomas')



1. **[0.6] Mostrar el id de las rutas que tienen más de un viaje registrado.**

SELECT r.idRuta FROM ruta r WHERE r.numeroViajes > 1



1. **[0.6] Obtener el nombre de los conductores que tienen viajes tanto a perisur como a barranca (a ambos, no sólo a uno u otro).**

SELECT c.IdCond FROM conductor c

INNER JOIN ruta r ON c.IdCond = r.idCond

INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta

INNER JOIN viaje v ON rv.idViaje = v.idViaje WHERE v.direccion = 'barranca' OR v.direccion = 'perisur' EXCEPT (SELECT c.IdCond FROM conductor c

INNER JOIN ruta r ON c.IdCond = r.idCond

INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta

INNER JOIN viaje v ON rv.idViaje = v.idViaje WHERE v.direccion = 'perisur')

UNION (SELECT c.IdCond FROM conductor c

INNER JOIN ruta r ON c.IdCond = r.idCond

INNER JOIN ruta\_viaje rv ON r.idRuta = rv.idRuta

INNER JOIN viaje v ON rv.idViaje = v.idViaje WHERE v.direccion = 'barranca')

