Teorica

Pregunta 1: Indica y explica con tus palabras dos de las características deseables para los sistemas de bases de datos.

1. Escalable
    a. Es cuando la base de datos puede aumentar el tamaño sin comprometer la funcionalidad de dicha base
2. Eficiente
    a. Asi como el nombre como el nombre indica la base de datos tendría que estar con los datos que especificos, no con datos basura o relleno, para mejorar el tiempo de buscada y la facilidad de consulta

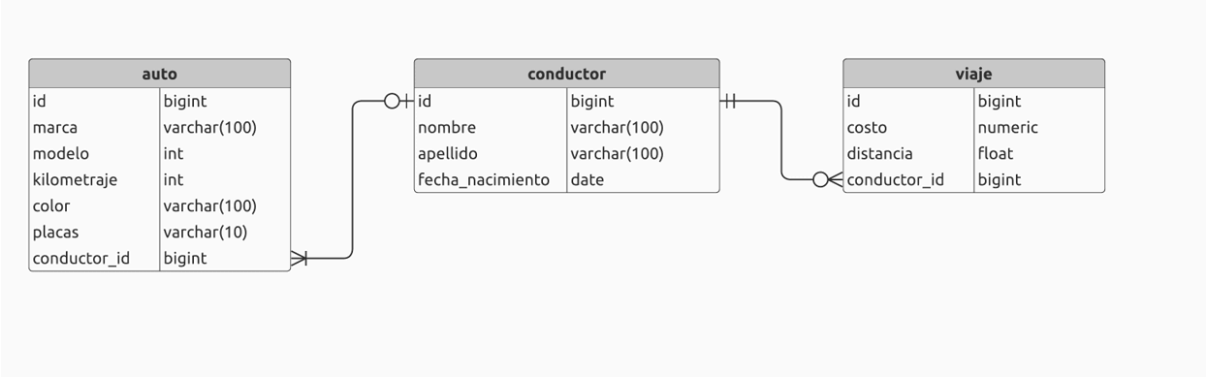
Pregunta 2: De acuerdo con el modelo relacional, indica el nombre de las llaves foráneas (foreign keys) para cada una de las siguientes relaciones o entidades

- Director_id
    - Al director.id
- Genero_id
    - Al genero.id

Pregunta 3: Explica con tus palabras cuál es la diferencia entre una llave primaria (primary key) y una llave foránea (foreign key)

1. Llave primaria
    a. Es el atributo en una relación donde se cumple que cada cada atributo es único para dicha relación
2. Llave foranea
    a. Es el conjunto que estable una correspondencia entre dos o mas tablas

Pregunta 4: De acuerdo al siguiente diagrama para una aplicación tipo Uber, explica brevemente (1) cómo se relaciona la entidad conductor con viaje y (2) cómo se relaciona la entidad conductor con auto. No olvides tomar en cuenta la cardinalidad y la naturaleza opcional u obligatoria de la interacción entre entidades.



- Conductor-viaje
    - La relación que se observa por la pata de cuervo es
        - Viaje - conductor
            - Un viaje tiene solo un conducto es único y obligatorio
        - Conductor - viaje
            - Un conductor puede tener mas de un viaje
            - Un conductor puede tener 0 viajes

- Conductor-auto
    - La relación que se observa por la pata de cuervo es
        - Conductor-auto
            - Un conductor puede tener varios autos
            - Es obligatorio tener conductor
        - Auto-conductor
            - Un auto solo tiene un conductor
            - Puede que no tenga conductor (opcional)

Practica

```
/*Ejercicio 1: Utilizando la tabla ruta, escribe
un query que regrese todas las tuplas de la tabla.
Ordena los resultados de forma que las rutas de
mayor distancia (mayor valor para la columna
millas) aparezcan primero. La relación resultante
tendrá las siguientes columnas:*/

SELECT ruta.ciudad_origen_id,
ruta.ciudad_destino_id, ruta.millas
   FROM ruta
   ORDER BY millas DESC;

/*Ejercicio 2: Utilizando la tabla pasajero_vuelo,
indica los ids de pasajeros que viajan con 2 o
más maletas (i.e., no_maletas). Favor de incluir
el número de maletas con las que viajan y el id
del vuelo. La relación resultante tendrá las
siguientes columnas:*/

SELECT pasajero_vuelo.pasajero_id,
pasajero_vuelo.no_maletas, pasajero_vuelo.vuelo_id
   FROM pasajero_vuelo
   WHERE no_maletas >= 2;

/* Ejercicio 3: Indica el id, nombre, apellido y
correo del pasajero (o los pasajeros) con el mayor
número de vuelos. La query se deberá asegurar que
en caso de un empate, se devuelva una
```

```
tupla por cada pasajero que comparta el número
máximo de vuelos. La relación resultante
tendrá las siguientes columnas: */

WITH pasajero_vuelos AS(
   SELECT pasajero.id, count(*) AS vuelos
       FROM pasajero
       JOIN pasajero_vuelo ON pasajero.id =
pasajero_vuelo.pasajero_id
       GROUP BY pasajero.id
)
SELECT pasajero.id, pasajero.nombre,
pasajero.apellido, pasajero.correo,
pasajero_vuelos.vuelos
   FROM pasajero_vuelos
   JOIN pasajero ON  pasajero.id =
pasajero_vuelos.id
   WHERE vuelos = (SELECT MAX(vuelos) FROM
pasajero_vuelos);


/* Ejercicio 4: Airitam quiere conocer la
popularidad de los destinos a los que viaja.
Proporciona
una query que regrese el id y nombre de las
ciudades así como el número de pasajeros que
han volado hacia ese destino. Ordene los
resultados de manera descendente en número de
pasajeros y como segundo criterio ordene con el
nombre de la ciudad de forma ascendente.
No confundir número de vuelos con número de
pasajeros. La relación resultante tendrá las
siguientes columnas. */
```

```sql
SELECT ruta.ciudad_destino_id, ciudad.nombre AS
ciudad_nombre, COUNT(pasajero_id) AS num_pasajeros
   FROM ruta
   JOIN ciudad ON ruta.ciudad_destino_id =
ciudad.id
   JOIN vuelo ON ruta.id = vuelo.ruta_id
   JOIN pasajero_vuelo ON vuelo.id =
pasajero_vuelo.vuelo_id
   GROUP BY ruta.ciudad_destino_id, ciudad.nombre
   ORDER BY num_pasajeros DESC, nombre;

/*Ejercicio 5: Airitam está preparando un
presupuesto para comprar aviones nuevos. Para
saber
cuántos aviones comprar, desea conocer aquellos
aviones que han viajado más de 100,000
(cien mil) millas. Escribe una query que indique
el id, modelo y fecha de fabricación de todos
aquellos aviones que han volado más de 100,000
(cien mil) millas.

Es importante notar que el
número de millas viajadas está dada por la suma de
las millas de todos los vuelos
efectuados por dicho avión. La relación resultante
tendrá las siguientes columnas*/


WITH suma_aviones AS (
   SELECT ruta.avion_id, SUM(millas) as suma
       FROM ruta
       JOIN vuelo ON ruta.id = vuelo.ruta_id
       GROUP BY avion_id
       HAVING SUM(millas) > 100000
       ORDER BY avion_id
```

```
)

SELECT DISTINCT ruta.avion_id, avion.modelo,
avion.fecha_fabricacion, suma_aviones.suma
   FROM suma_aviones
   JOIN ruta ON ruta.avion_id =
suma_aviones.avion_id
   JOIN avion ON avion.id = suma_aviones.avion_id
   ORDER BY avion_id;

/* Ejercicio 6: Airitam quiere identificar todas
los rutas que puede ofrecer como un boleto
redondo, es decir, que puede ofrecer como un
paquete con el vuelo de ida y el de regreso.
Favor de escribir un query que devuelva los pares
de ciudades (ciudad_1_id, ciudad_2_id) que
pueda ofrecer como boleto redondo.  La relación
resultante tendrá las siguientes columnas. */



SELECT r1.ciudad_origen_id AS ciudad_1_id,
r1.ciudad_destino_id AS ciudad_2_id
   FROM ruta AS r1
   JOIN ruta AS r2 ON r1.ciudad_origen_id =
r2.ciudad_destino_id
                          AND r1.ciudad_destino_id
= r2.ciudad_origen_id
   WHERE r1.ciudad_origen_id >
r1.ciudad_destino_id;
```