Mariana Cediel – 202321548

Alejandro Cruz – 201912149

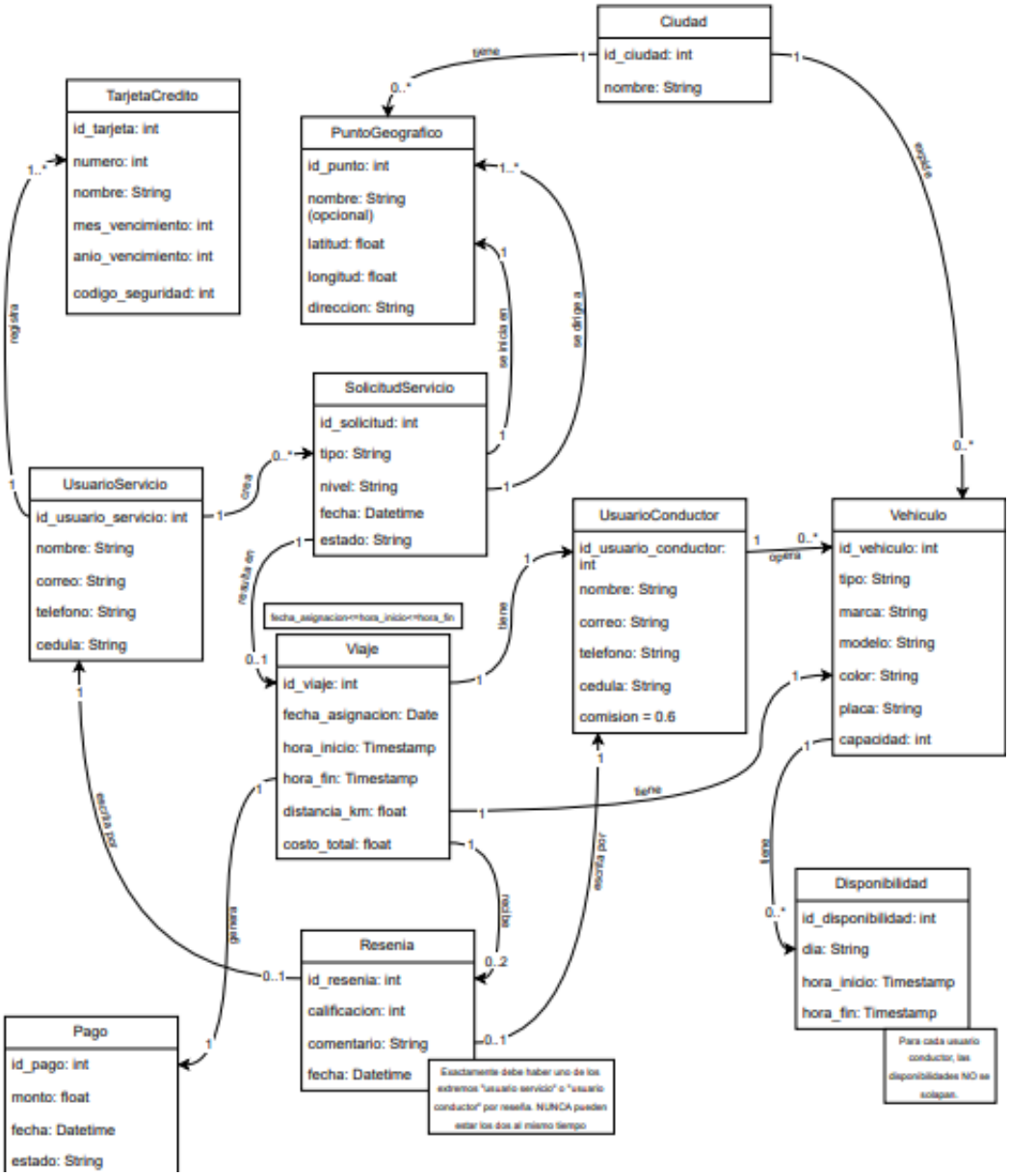
Nicolás Hernández - 202322148

Sistemas Transaccionales

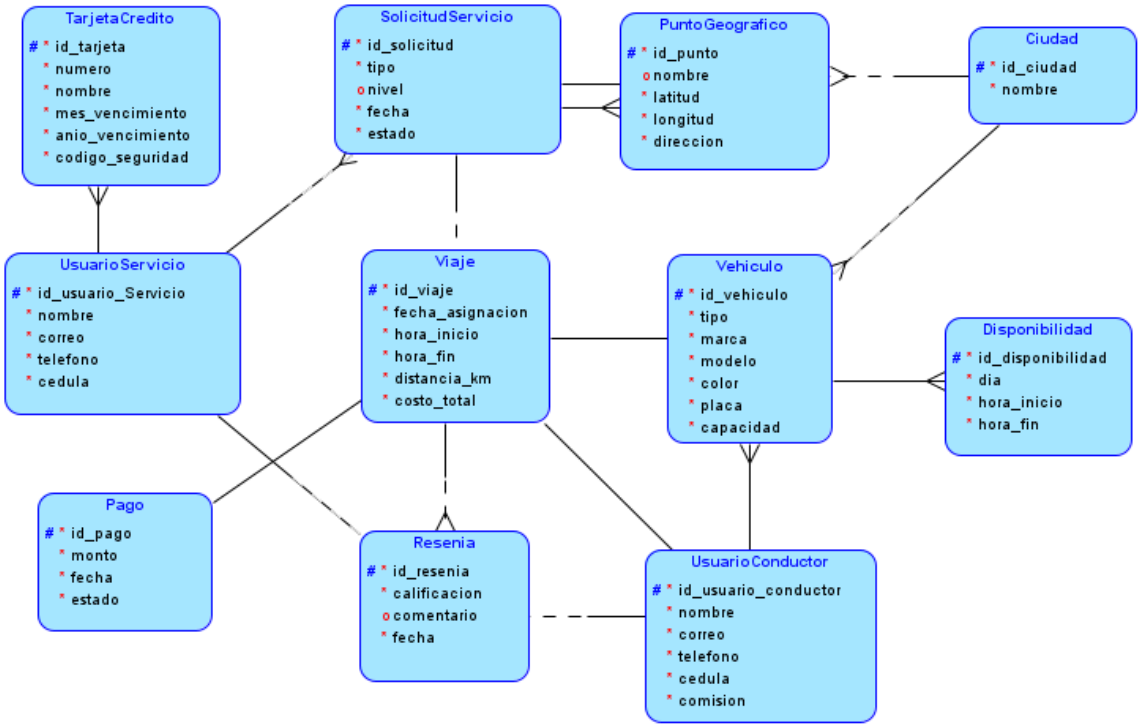
Entrega 1 - Diseño

(45%) Análisis y modelo conceptual

1. **(10%)** Proponga un modelo conceptual en UML que represente el negocio de AlpesCab.



2. **(35%)** Proponga un modelo conceptual en E/R que describa las entidades del modelo de datos para la aplicación que se quiere desarrollar. Para ello use Data Modeler.



(45%) Diseño de la base de datos

3. **(25%)** Desarrolle el modelo de datos relacional correspondiente al modelo conceptual UML propuesto. Este modelo debe quedar descrito en la plantilla de Excel disponible para ello en BN. Debe justificar la selección de las tablas a través del proceso de trasformación de UML a relacional visto en clase (algoritmo modificado de Chen).

Tablas de entidades

1. **Ciudad**

Ciudad	
id_ciudad (NUMBER)	nombre (VARCHAR2(250CHAR))
PK, NN, ND, SA	NN

1	Bogotá
2	Medellín
3	Cali
4	Barranquilla
5	Bucaramanga

2. Tarjeta de crédito

TarjetaCredito					
id_tarjeta (NUMBER)	numero (INTEGER)	nombre (VARCHAR2(250 CHAR))	mes_vencimiento (INTEGER)	anio_vencimiento (INTEGER)	codigo_seguridad (INTEGER)
PK, NN, ND, SA	NN, UA	NN, UA	NN, CK, UA	NN, CK, UA	NN, UA
1	10000001	Juan P. González	1	2027	123
2	10000002	María L. Sánchez	3	2028	456
3	10000003	Carlos A. Pérez	5	2029	789
4	10000004	Ana R. Torres	7	2030	321
5	10000005	Jorge M. Rodríguez	9	2031	654

3. Punto geográfico

PuntoGeografico				
id_punto (NUMBER)	nombre (VARCHAR2(250 CHAR))	latitud (FLOAT)	longitud (FLOAT)	direccion (VARCHAR2(250 CHAR))
PK, NN, ND, SA	NN, UA	NN	NN	NN
1001	Oficina Norte	4.711	-740.721	Av. 7 #123-45, Bogotá
1002	Centro Comercial Andino	46.687	-740.544	Cra. 11 #82-71, Bogotá
1003	Parque Lleras	62.088	-75.566	El Poblado, Medellín
1004	Aeropuerto El Dorado	47.016	-741.469	Calle 26 #103-9, Bogotá
1005	Terminal Cali	34.516	-765.319	Cl. 30N #2N-29, Cali

4. Solicitud de servicio

SolicitudServicio				
id_solicitud (NUMBER)	tipo (VARCHAR2(250 CHAR))	nivel (VARCHAR2(50CHAR))	fecha (DATE)	estado (VARCHAR2(250 CHAR))
PK, NN, ND, SA	NN, CK	CK	NN	NN, CK
2001	pasajeros	estandar	30/08/2025	creada
2002	comida		30/08/2025	asignada
2003	mercancías		31/08/2025	creada
2004	pasajeros	confort	01/09/2025	cancelada
2005	pasajeros	large	02/09/2025	asignada

- Estado: "creada", "asignada", "cancelada".

- Tipo: "pasajeros", "comida", "mercancías".

5. Usuario de servicio

UsuarioServicio				
id_usuario_servicio (NUMBER)	nombre (VARCHAR2(250 CHAR))	correo (VARCHAR2(250CHAR))	telefono (VARCHAR2 (50CHAR))	cedula (VARCHAR2 (50 CHAR))
PK, NN, ND, SA	NN	NN, ND	NN	NN, ND
3001	Laura Rincón	laura.rincon@example.com	3001112233	1012345678
3002	David Mora	david.mora@example.com	3002223344	1012345679
3003	Sofía Martínez	sofia.martinez@example.com	3003334455	1012345680
3004	Andrés Gómez	andres.gomez@example.com	3004445566	1012345681
3005	Valentina Ruiz	valentina.ruiz@example.com	3005556677	1012345682

6. Vehículo

Vehiculo						
id_vehiculo (NUMBER)	tipo (VARCHAR2 (250CHAR))	marca (VARCHAR2 (250CHAR))	modelo (VARCHAR2 (250CHAR))	color (VARCHAR2 (250CHAR))	placa (VARCHAR2 (50CHAR))	capacidad (NUMBER)
PK, NN, ND, SA	NN, CK	NN, UA	NN, UA	NN, ND, UA	NN, UA	NN

4001	carro	Toyota	Corolla	Rojo	ABC123	5
4002	camioneta	Renault	Duster	Azul Marino	DEF456	5
4003	carro	Chevrolet	Onix	Blanco Perla	GHI789	5
4004	carro	Mazda	CX-30	Negro	JKL234	5
4005	motocicleta	Honda	CBR500	Gris Plata	MNO567	2

- Tipo: "carro", "camioneta", "motocicleta".

7. Usuario conductor

UsuarioConductor					
id_usuario_conductor (NUMBER)	nombre (VARCHAR2(250CHAR))	correo (VARCHAR2(250CHAR))	telefono (VARCHAR2(50CHAR))	cedula (VARCHAR2(50CHAR))	comision (FLOAT)
PK, NN, ND, SA	NN, UA	NN, ND, UA	NN, ND, UA	NN, ND, UA	NN
5001	Carlos Gómez	carlos.gomez@alfa.com	3111111111	1023456700	0.6
5002	Ana Ruiz	ana.ruiz@alfa.com	3122222222	1023456701	0.6
5003	Luis Pérez	luis.perez@alfa.com	3133333333	1023456702	0.6
5004	María Díaz	maria.diaz@alfa.com	3144444444	1023456703	0.6
5005	Jorge Torres	jorge.torres@alfa.com	3155555555	1023456704	0.6

8. Viaje

Viaje					
id_viaje (NUMBER)	fecha_asignacion (DATE)	hora_inicio (TIMESTAMP WITH LOCAL TIME ZONE)	hora_fin (TIMESTAMP WITH LOCAL TIME ZONE)	distancia_km (FLOAT)	costo_total (FLOAT)
PK, NN, ND, SA	NN, SA	NN, SA	NN, SA	NN, SA	NN, SA
6001	30/08/2025	30/08/2025 8:15	30/08/2025 8:45	7,7	18000
6002	30/08/2025	30/08/2025 9:10	30/08/2025 9:55	10,2	24500
6003	31/08/2025	31/08/2025 18:00	31/08/2025 18:20	4	9600
6004	01/09/2025	01/09/2025 7:30	01/09/2025 8:20	12,3	29500

6005	02/09/2025	02/09/2025 22:05	02/09/2025 22:35	8,1	19500

- hora_inicio <= hora_fin

9. Disponibilidad

Disponibilidad			
id_disponibilidad (NUMBER)	dia (VARCHAR2(50 CHAR))	hora_inicio (TIMESTAMP WITH LOCAL TIME ZONE)	hora_fin (TIMESTAMP WITH LOCAL TIME ZONE)
PK, NN, ND, SA	NN, CK	NN, UA	NN, UA
7001	lunes	01/09/2025 6:00	01/09/2025 9:00
7002	martes	02/09/2025 8:00	02/09/2025 12:00
7003	miercoles	03/09/2025 10:00	03/09/2025 14:00
7004	jueves	04/09/2025 12:00	04/09/2025 18:00
7005	viernes	05/09/2025 14:00	05/09/2025 20:00

- Dia: "lunes", "martes", "miercoles", "jueves", "viernes", "sabado", "domingo"
- hora_inicio <= hora_fin

10. Reseña

Resenia			
id_resenia (NUMBER)	calificacion (INTEGER)	comentario (VARCHAR2(1000 CHAR))	fecha (DATE)
PK, NN, ND, SA	NN, CK	UA	NN, SA
8001	5	Excelente servicio, muy puntual.	30/08/2025
8002	4	Buen trato y conducción segura.	30/08/2025
8003	3	Correcto, pero podría mejorar.	31/08/2025
8004	2	Retraso en la recogida.	01/09/2025
8005	1	Mala experiencia en general.	02/09/2025

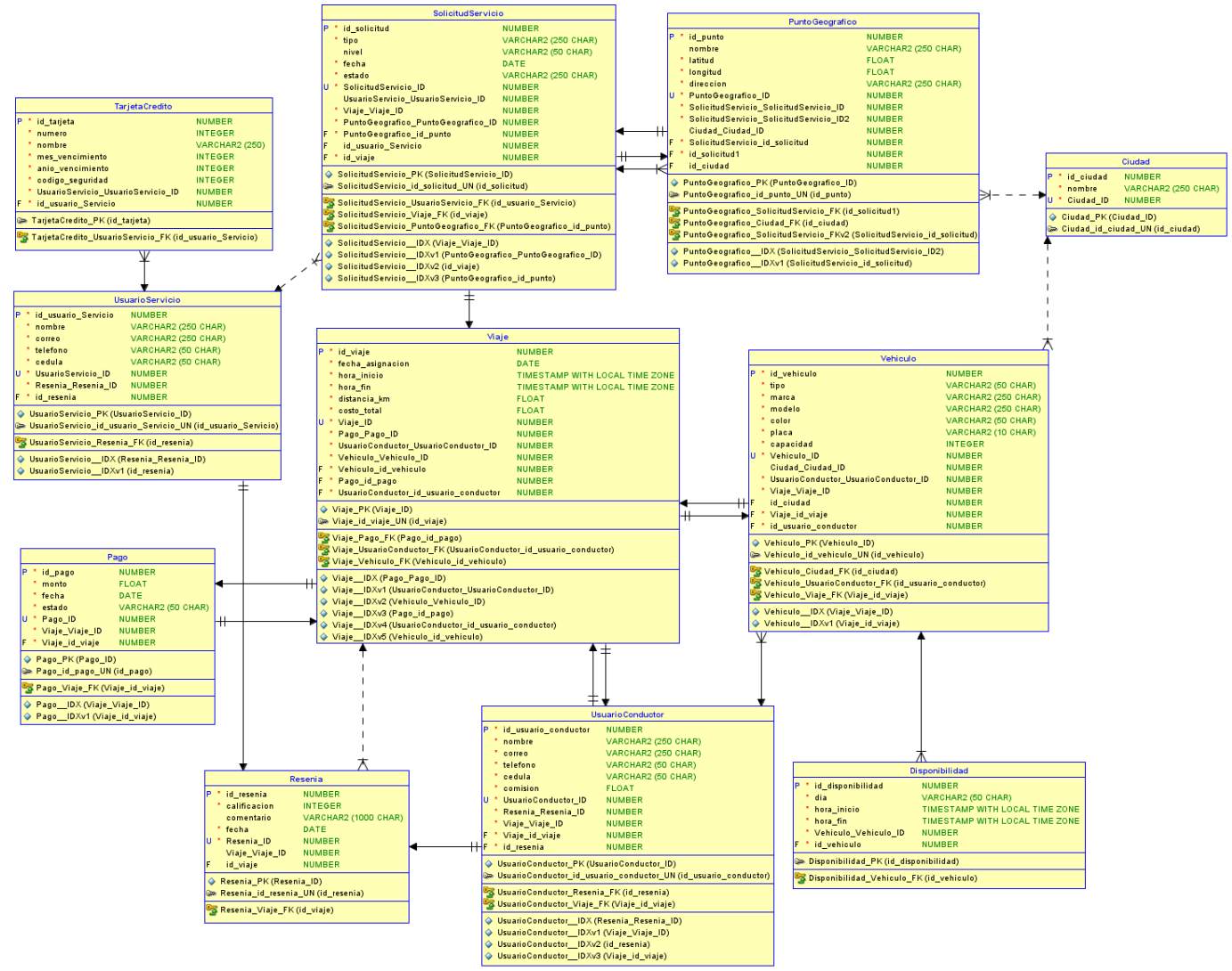
- Calificación: 1, 2, 3, 4, 5

11. Pago

Pago			
id_pago (NUMBER)	monto (FLOAT)	fecha (DATE)	estado (VARCHAR2(50 CHAR))
PK, NN, ND, SA	NN, SA	NN, SA	NN, CK
9001	18000	30/08/2025	completado
9002	24500	30/08/2025	en espera
9003	9600	31/08/2025	completado
9004	29500	01/09/2025	rechazado
9005	19500	02/09/2025	completado

- Estado: "completado", en espera", "rechazado"

Modelo relacional



Tablas relacionales

12. Usuario servicio registra 1 o más tarjetas

TarjetaCredito			
id_tarjeta (NUMBER)	numero (INTEGER)	nombre (VARCHAR2(250CHAR))	mes_vencimiento (INTEGER)
PK, NN, ND, SA	NN, UA	NN, UA	NN, CK, UA
1	10000001	Juan P. González	1
2	10000002	María L. Sánchez	3
3	10000003	Carlos A. Pérez	5
4	10000004	Ana R. Torres	7
5	10000005	Jorge M. Rodríguez	9

anio_vencimiento (INTEGER)	codigo_seguridad (INTEGER)	id_usuario_servicio (NUMBER)
NN, CK, UA	NN, UA	NN, FK$_{\text{UsuarioServicio.id_usuario_servicio}}$
2027	123	3001
2028	456	3002
2029	789	3003
2030	321	3004
2031	654	3005

(Es la misma tabla, la separamos para que se viera bien).

Justificación (algoritmo de Chen): la PK del lado con cardinalidad 1 (en este caso usuario de servicio), se incorpora como FK en el lado de cardinalidad N (tarjeta de crédito).

13. Ciudad puede tener muchos puntos geográficos

PuntoGeografico			
id_punto (NUMBER)	nombre (VARCHAR2(250CHAR))	latitud (FLOAT)	longitud (FLOAT)
PK, NN, ND, SA	NN, UA	NN	NN
1001	Oficina Norte	4.711	-740.721

1002	Centro Comercial Andino	46.687	-740.544
1003	Parque Lleras	62.088	-75.566
1004	Aeropuerto El Dorado	47.016	-741.469
1005	Terminal Cali	34.516	-765.319

direccion (VARCHAR2(250CHAR))	id_ciudad (NUMBER)
NN	NN, FKCiudad.id_ciudad
Av. 7 #123-45, Bogotá	1
Cra. 11 #82-71, Bogotá	1
El Poblado, Medellín	2
Calle 26 #103-9, Bogotá	1
Cl. 30N #2N-29, Cali	3

(Es la misma tabla, la separamos para que se viera bien).

Justificación (algoritmo de Chen): la PK del lado con cardinalidad 1 (ciudad), se incorpora como FK en el lado de cardinalidad N (Punto Geografico).

14. Un usuario solicita un servicio y un servicio tiene un punto de partida

SolicitudServicio			
id_solicitud (NUMBER)	tipo (VARCHAR2(250CHAR))	nivel (VARCHAR2(50CHAR))	fecha (DATE)
PK, NN, ND, SA	NN, CK	CK	NN
2001	pasajeros	estandar	30/08/2025
2002	comida		30/08/2025
2003	mercancías		31/08/2025
2004	pasajeros	confort	01/09/2025
2005	pasajeros	large	02/09/2025

estado (VARCHAR2(250CHAR))	id_usuario_servicio (NUMBER)	id_punto_partida (NUMBER)

NN, CK	NN, FK UsuarioServicio.id_usuario_servicio	NN, FK PuntoGeografico.id_punto
creada	3001	1001
asignada	3002	1002
creada	3003	1003
cancelada	3004	1004
asignada	3005	1005

(Es la misma tabla, la separamos para que se viera bien).

- Justificación (algoritmo de Chen): la PK del lado con cardinalidad 1 (usuario de servicio), se incorpora como FK en el lado de cardinalidad N (Solicitud de servicio).
- Justificación (algoritmo de Chen): la relación es de 1 a 1 entre Solicitud de servicio y punto geográfico. la PK de uno de los dos lados (punto geográfico), se incorpora como FK en el otro lado (Solicitud de servicio).

15. Vehículo tiene una ciudad de expedición y un usuario conductor puede tener varios vehículos.

Vehiculo			
id_vehiculo (NUMBER)	tipo (VARCHAR2(250CHAR))	marca (VARCHAR2(250CHAR))	modelo (VARCHAR2(250CHAR))
PK, NN, ND, SA	NN, CK	NN, UA	NN, UA
4001	carro	Toyota	Corolla
4002	camioneta	Renault	Duster
4003	carro	Chevrolet	Onix
4004	carro	Mazda	CX-30
4005	motocicleta	Honda	CBR500

color (VARCHAR2(250CHAR))	placa (VARCHAR2(50CHAR))	capacidad (NUMBER)	id_usuario_conductor (NUMBER)

NN, ND, UA	NN, UA	NN	NN, FKUsuarioConductor.id_usuario_conductor
Rojo	ABC123	5	5001
Azul Marino	DEF456	5	5002
Blanco Perla	GHI789	5	5003
Negro	JKL234	5	5004
Gris Plata	MNO567	2	5005

id_ciudad_expedicion (NUMBER)
NN, FKCiudad.id_ciudad
1
2
3
4
5

(Es la misma tabla, la separamos para que se viera bien).

- Justificación (algoritmo de Chen): la PK del lado con cardinalidad 1 (usuario conductor), se incorpora como FK en el lado de cardinalidad N (vehículo).
- Justificación (algoritmo de Chen): la relación es de 1 a 1 entre ciudad y vehículo. la PK de uno de los dos lados (ciudad), se incorpora como FK en el otro lado (vehículo).

16. A un viaje se le asigna un usuario conductor, cada viaje tiene un vehículo y un viaje tiene un punto de partida

Viaje			
id_viaje (NUMBER)	fecha_asignacion (DATE)	hora_inicio (TIMESTAMP WITH LOCAL TIME ZONE)	hora_fin (TIMESTAMP WITH LOCAL TIME ZONE)

PK, NN, ND, SA	NN, SA	NN, SA	NN, SA
6001	30/08/2025	30/08/2025 8:15	30/08/2025 8:45
6002	30/08/2025	30/08/2025 9:10	30/08/2025 9:55
6003	31/08/2025	31/08/2025 18:00	31/08/2025 18:20
6004	01/09/2025	01/09/2025 7:30	01/09/2025 8:20
6005	02/09/2025	02/09/2025 22:05	02/09/2025 22:35

distancia_km (FLOAT)	costo_total (FLOAT)	id_usuario_conductor (NUMBER)	id_vehiculo (NUMBER)
NN, SA	NN, SA	NN, FKUsuarioConductor.id_usuario_conductor	NN, FKVehiculo.id_vehiculo
7,7	18000	5001	4001
10,2	24500	5002	4002
4	9600	5003	4003
12,3	29500	5004	4004
8,1	19500	5005	4005

id_punto_partida (NUMBER)
NN, FKPuntoGeografico.id_punto
1001
1002
1003
1004
1005

(Es la misma tabla, la separamos para que se viera bien).

- Justificación (algoritmo de Chen): la relación es de 1 a 1 entre ciudad y vehículo. la PK de uno de los dos lados (vehiculo), se incorpora como FK en el otro lado (viaje).
- Justificación (algoritmo de Chen): la relación es de 1 a 1 entre ciudad y vehículo. la PK de uno de los dos lados (usuario conductor), se incorpora como FK en el otro lado (viaje).
- Justificación (algoritmo de Chen): la relación es de 1 a 1 entre ciudad y vehículo. la PK de uno de los dos lados (punto geografico), se incorpora como FK en el otro lado (viaje).

Justificación general del proceso

El modelo relacional fue construido aplicando sistemáticamente el **algoritmo modificado de Chen** para la transformación UML → Relacional.

Con estas reglas se garantizó que el paso de UML a relacional preserve las cardinalidades, dependencias y restricciones de negocio descritas en el caso ALPESCAB

Justificación de las tablas						
Relacion	**Origen del UML**	**Regla Chen aplicada**	**PK**	**FK**	**Restricciones**	**Justificación**
Ciudad	Clase fuerte independiente	R1 (Clase independiente → Tabla)	id_ciudad	—	nombre único	Se requiere como catálogo base de ciudades; es entidad fuerte sin dependencia existencial. Necesaria para identificar placas de vehículos y localización de puntos geográficos.
UsuarioServicio	Subclase de Usuario (especialización)	R7 (Generalización → Tabla por subclase)	id_usuario_servicio	—	correo único, cedula única	Actor que solicita servicios y maneja tarjetas de crédito. Tiene atributos propios (correo, cédula únicas). Se separa de Conductor por roles y restricciones diferentes.

UsuarioConductor	Subclase de Usuario (especialización)	R7 (Generalización → Tabla por subclase)	id_usuario_conductor	Viaje, Resenia según tu modelo	comision en rango	Actor que presta servicios. Tiene atributo específico (comisión) y participa en relaciones distintas (Vehículo, Disponibilidad, Viaje, Reseña). Se justifica tabla propia.
Vehiculo	Clase fuerte asociada a Conductor y Ciudad	R1 + R2 (Entidad → Tabla, Relación 1:N → FK en el lado N)	d_vehiculo	Ciudad.id_ciudad, UsuarioConductor.id_usuario_conductor	placa única, dominio de tipo y nivel	Entidad independiente con PK propia. Incluye FKs a UsuarioConductor y Ciudad para reflejar que cada vehículo pertenece a un conductor y a una ciudad.
Disponibilidad	Asociación UML entre Conductor y Vehículo con atributos	R2 (Relación 1:N → FK en el lado N)	PK: id_disponibilidad	FKs: Vehiculo.id_vehiculo	rango horario válido	Se convierte en tabla propia porque tiene atributos adicionales (día, hora_inicio, hora_fin, tipo_servicio). No podía representarse solo como FK simple.
PuntoGeografico	Clase fuerte dependiente de Ciudad	R1 + R2 (Entidad → Tabla, Relación 1:N → FK en el lado N)	id_punto	Ciudad.id_ciudad, SolicitudServicio_SolicitudServicio_ID, SolicitudServicio_SolicitudServicio_ID2	latitud y longitud obligatorias	Entidad necesaria para modelar orígenes y destinos. Incluye atributos de latitud, longitud y dirección. Lleva FK a Ciudad.
SolicitudServicio	Clase central del flujo UML	R1 + R2 (Entidad → Tabla, Relación 1:N	id_solicitud	UsuarioServicio, PuntoGeo	dominios de tipo, nivel, estado	Representa la orden generada por el cliente. Incluye FKs a UsuarioServicio y

		→ FK en el lado N)		grafico, Viaje		PuntoGeografico (origen/destino). Puede derivar en un Viaje (0..1).
Viaje	Clase dependiente de SolicitudServicio	R8 (Entidad dependiente → Tabla con FK obligatoria)	id_viaje	UsuarioConductor, Vehiculo, Pago	tiempos coherentes, costo_total, distancia_km	El viaje es la materialización de la solicitud. Se convierte en tabla con PK propia y FKs hacia SolicitudServicio, UsuarioConductor y Vehiculo.
Pago	Clase dependiente de Viaje	R8 (Entidad dependiente → Tabla con FK obligatoria)	id_pago	Viaje	estado de pago	La existencia del pago depende del viaje. Se convierte en tabla con PK propia y FK hacia Viaje. Representa la transacción financiera del servicio.
Resenia	Clase dependiente de Viaje	R8 (Entidad dependiente → Tabla con FK obligatoria)	id_resenia	Viaje	calificacion en rango	La reseña depende de la existencia del viaje. Se convierte en tabla con PK propia y FK hacia Viaje. Permite evaluación mutua conductor–cliente.
TarjetaCredito	Clase dependiente de UsuarioServicio	R8 (Entidad dependiente → Tabla con FK obligatoria)	id_tarjeta	UsuarioServicio	dominios de mes y año, seguridad	La tarjeta depende de un UsuarioServicio. Se convierte en tabla para gestionar medios de pago asociados a clientes, cumpliendo con RF2.

4. **(20%)** Determine el nivel de normalización en que se encuentra su modelo. Para ello adjunte a la entrega la verificación de las formas normales vistas en clase para

cada relación. Aviso: si su modelo no está en FN BC es posible que no pueda implementar todos los requerimientos de la aplicación.

Nuestro modelo se encuentra en el nivel de normalización BCNF, es decir, cumple con la 1NF, la 2NF, la 3NF y la BCNF.

Para verificar que nuestro modelo cumple con la 1NF, debimos verificar en cada relación que:

- Todos los valores en las tablas son atómicos, es decir, cada celda de las tablas contienen un solo valor.
- Cada columna de la tabla tiene un nombre único.

Para verificar que nuestro modelo cumple con la 2NF, debimos verificar en cada relación que:

- Cada atributo en una tabla depende completamente de toda la llave primaria y no de una parte de ella.
- Los datos se organizan de tal manera que no hay dependencias parciales de los atributos no clave en la llave primaria.

Para verificar que nuestro modelo cumple con la 3NF, debimos verificar en cada relación que:

- No hay dependencias transitivas entre los atributos no clave y la llave primaria. Esto significa que ningún atributo no clave depende de otro atributo no clave a través de la llave primaria.

Para verificar que nuestro modelo cumple con la BCNF, debimos verificar en cada relación que:

- El modelo cumple con la 3NF y todas las llaves son simples.

(10%) Escenarios de prueba

5. Diseñe escenarios que les permitirán probar los RF (ver secciones de Requerimientos funcionales del documento marco del caso de estudio (RF1-RF11 y RFC1-RFC4) y asegurar la corrección y calidad de los datos en la base de datos.

- Roles USR cliente = UsuarioServicio, COND = UsuarioConductor, ADM = Administrador
- Datos base para precondiciones:
    - Ciudad: (1, Bogotá), (2, Medellín)

- UsuarioServicio: 1001 Carlos, 1002 Laura
- UsuarioConductor: 2001 Ana, 2002 Luis
- Vehiculo: 3001 de Ana en Bogotá placa ABC123, 3002 de Luis en Medellín placa XYZ789
- PuntoGeografico: 4001 Aeropuerto BOG en Bogotá, 4002 Parque Poblado en Medellín

1. Pruebas de corrección y calidad de datos

1.1 Unicidad de tuplas

ID	Objetivo	Tabla	Pre condiciones	Datos de entrada	Pasos	Resultado esperado
UNI 1	Validar PK única	Ciudad	Existe id_ciudad 1	Insertar Ciudad con id_ciudad 1	ADM intenta crear registro con id repetido	Rechazo por violar PK
UNI 2	Validar Unique por negocio	Usuario Servicio	Existe correo carlos@mail.com	Crear UsuarioServicio con mismo correo	ADM intenta crear registro	Rechazo por violar Unique correo
UNI 3	Validar placa única	Vehiculo	Existe placa ABC123	Crear Vehiculo con placa ABC123	ADM intenta crear registro	Rechazo por violar Unique placa

1.2 Integridad referencial con FKs

ID	Objetivo	Tabla hija	FK	Pre condiciones	Caso	Pasos	Resultado esperado

FK 1	FK valida hacia Ciudad	Vehiculo	Ciudad_id_ciudad	Ciudad 1 existe	Éxito	Crear Vehiculo con Ciudad_id_ciudad 1	Inserción aceptada
FK 2	FK inválida hacia Ciudad	Vehiculo	Ciudad_id_ciudad	Ciudad 99 no existe	Falla	Crear Vehiculo con Ciudad_id_ciudad 99	Rechazo por FK
FK 3	FK valida hacia UsuarioConductor	Vehiculo	Usuario Conductor_id_usuario_conductor	Conductor 2001 existe	Éxito	Crear Vehiculo de 2001	Inserción aceptada
FK 4	FK inválida hacia Vehiculo	Disponibilidad	Vehiculo_Vehiculo_ID	Vehiculo 3001 existe, 3999 no	Éxito y falla	Crear disp con 3001 y con 3999	Acepta primera, rechaza segunda
FK 5	FK valida hacia Viaje	Resenia	Viaje_Viaje_ID	Viaje 5001 existe	Éxito	Crear Reseña para 5001	Inserción aceptada
FK 6	FK inválida hacia Viaje	Pago	Viaje_Viaje_ID	Viaje 5999 no existe	Falla	Crear Pago con 5999	

1.3 Restricciones de chequeo

ID	Objetivo	Tabla	Check esperado	Datos de entrada	Resultado esperado
CK 1	Comisión en rango	Usuario Conductor	0 a 1	comision = 1.2	Rechazo por check
CK 2	Calificación válida	Resenia	0 a 5	calificacion = 7	Rechazo por check

CK 3	Rango horario válido	Disponibilidad	hora_fin > hora_inicio	inicio 10:00 fin 09:00	Rechazo por check
CK 4	Día válido	Disponibilidad	día de catálogo o dominio	día = “D8” si el dominio es controlado	Rechazo por check o catálogo

1.4 Reglas de negocio clave

ID	Objetivo	Regla	Tablas	Precondiciones	Caso	Resultado esperado
RB 1	No traslape de disponibilidad por vehículo o conductor	Sin solapes para mismo recurso y franja	Disponibilidad	Ya existe Lunes 08:00 a 12:00 para 3001	Intentar crear Lunes 10:00 a 11:00 para 3001	Rechazo por trigger de solape
RB 2	Consistencia de tiempos de viaje	hora_fin >= hora_inicio	Viaje	Asignado 14:05	Registrar fin 13:55	Rechazo por check de tiempo
RB 3	Un pago por viaje o por solicitud según tu modelo	Unicidad lógica	Pago y Viaje	Existe Pago para 5001	Crear otro Pago para 5001	Rechazo por unique o validación de aplicación
RB 4	Datos mínimos para solicitud	Tipo, fecha, estado, origen presentes	SolicitudServicio, PuntoGeografico	UsuarioServicio 1001 y Punto 4001 existen	Crear solicitud sin tipo o sin origen	Rechazo por NN o validación

2. Pruebas funcionales por requerimiento RF1 a RF11

Cada RF con un caso exitoso y un caso de falla. Los "pasos" se describen a nivel de operación de aplicación y verificación en BD.

RF1 Registrar ciudad

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF1 OK	Crear ciudad válida	ADM	Ingresar nombre "Barranquilla" y guardar	Ciudad creada, nombre único visible en catálogo
RF1 FAIL	Duplicado de nombre	ADM	Intentar crear "Bogotá" nuevamente	Rechazo por ND o mensaje de duplicado

RF2 Registrar UsuarioServicio

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF2 OK	Registrar cliente	USR	Completar nombre, correo, teléfono, cédula	UsuarioServicio creado con correo y cédula únicos
RF2 FAIL	Correo duplicado	USR	Registrar con correo de 1001	Rechazo por ND correo

RF3 Registrar UsuarioConductor

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF3 OK	Registrar conductor	COND	Completar datos y comision 0.6	Conductor creado
RF3 FAIL	Comisión inválida	COND	comision 1.5	Rechazo por check de comisión

RF4 Registrar Vehiculo

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF4 OK	Vehículo del conductor	COND	Seleccionar conductor 2001 y ciudad 1. Placa ABC999	Vehículo creado. Placa única
RF4 FAIL	Ciudad inexistente	COND	Usar Ciudad 99	Rechazo por FK a Ciudad

RF5 Registrar disponibilidad de conductor

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF5 OK	Crear franja sin solape	COND	Lunes 08:00 a 12:00 para 3001	Disponibilidad creada
RF5 FAIL	Solape	COND	Crear Lunes 10:00 a 11:00 para 3001	Rechazo por trigger de solape

RF6 Registrar que un conductor está disponible para un tipo de servicio

Caso	Objetivo	Rol	Pasos	Resultado esperado

RF6 OK	Tipo PASAJEROS	COND	Marcar tipo_servicio PASAJEROS en disponibilidad	Disponibilidad guardada
RF6 FAIL	Tipo inválido	COND	tipo_servicio “AEREO”	Rechazo por check de dominio

RF7 Registrar puntos geográficos

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF7 OK	Crear punto en ciudad válida	USR	Latitud y longitud válidas. Ciudad 1	Punto creado
RF7 FAIL	Ciudad inexistente	USR	Ciudad 99	Rechazo por FK

RF8 Solicitar servicio y asignación

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF8 OK	Crear solicitud válida y asignar	USR	Tipo PASAJEROS nivel ESTANDAR, origen 4001. App asigna conductor cercano disponible 2001 con 3001	Solicitud en estado ASIGNADA y Viaje creado con fecha_asignacion
RF8 FAIL	Sin disponibilidad	USR	Mismo origen y hora sin ninguna disponibilidad para tipo	Solicitud queda CREADA o RECHAZADA según lógica. Sin Viaje asignado

RF9 Registrar viaje terminado

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF9 OK	Cerrar viaje con métricas	COND	Iniciar 14:05 fin 14:45 distancia 12.8 costo calculado	Viaje con hora_fin y costo_total persistidos
RF9 FAIL	Tiempos incoherentes	COND	Fin menor que inicio	Rechazo por check tiempos

RF10 Reseña del cliente al conductor

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF10 OK	Calificar viaje	USR	calificacion 5 comentario	Reseña creada ligada a Viaje
RF10 FAIL	Calificación inválida	USR	calificacion 6	Rechazo por check calificación

RF11 Reseña del conductor al cliente

Caso	Objetivo	Rol	Pasos	Resultado esperado
RF11 OK	Calificar viaje	COND	calificacion 4 comentario	Reseña creada ligada a Viaje
RF11 FAIL	Duplicado por política	COND	Intentar dos reseñas para el mismo viaje y autor	Rechazo por política de negocio si se aplica única por autor y viaje

3. Pruebas de consultas RFC1 a RFC4

Para cada RFC define datos de entrada, pasos y verificación del resultado esperado en términos de conteos, órdenes y porcentajes. No es necesario escribir SQL aquí, solo el criterio de aprobación.

RFC1 Histórico de servicios por usuario

ID	Objetivo	Datos de entrada	Pasos	Resultado esperado
RFC 1 A	Histórico UsuarioServicio	Usuario 1001 con solicitudes y viajes	Ejecutar consulta de histórico para 1001	Lista de solicitudes con estado y viajes asociados ordenados por fecha descendiente
RFC 1 B	Histórico UsuarioConductor	Conductor 2001 con viajes	Ejecutar consulta de histórico para 2001	Lista de viajes con fechas y costos ordenados por fecha

RFC2 Top 20 conductores

ID	Objetivo	Datos de entrada	Pasos	Resultado esperado
RFC2 A	Ranking por número de viajes	Viajes varios por 2001 y 2002	Agrupar por conductor y contar viajes, ordenar desc, limitar 20	2001 aparece por encima de 2002 si tiene más viajes
RFC2 B	Empates y bordes	20 o más conductores	Ejecutar ranking	Lista con 20 primeras filas sin sobrepasar límite, empates resueltos por segundo criterio si aplica

RFC3 Dinero ganado por vehículo y servicio

ID	Objetivo	Datos de entrada	Pasos	Resultado esperado
RFC 3 A	Suma por vehículo	Viajes de 3001 y 3002 con costo_total y comisión	Agrupar por id_vehiculo y tipo, sumar costo_total y comisión	Totales correctos por vehículo y tipo
RFC 3 B	Rango de fechas	Fechas cruzadas	Filtrar por rango	Sumas se ajustan al rango dado

RFC4 Utilización por ciudad y fechas

ID	Objetivo	Datos de entrada	Pasos	Resultado esperado
RFC4 A	Porcentajes por tipo y nivel	Viajes en Bogotá y Medellín en rango	Contar servicios por tipo y nivel en la ciudad y rango, calcular porcentaje sobre total de ese rango	Tabla ordenada del más usado al menos usado con porcentajes que suman 100 por ciento
RFC4 B	Caso sin datos	Ciudad sin viajes en rango	Ejecutar consulta	Devuelve cero filas o totales en cero sin error

Criterios de aceptación generales

- Toda inserción que viole PK, Unique, FK o Check debe ser rechazada por la base de datos o por la capa de aplicación antes de llegar a la BD.
- Los triggers y checks de negocio deben impedir estados imposibles: solapes de disponibilidad, tiempos de viaje invertidos.
- Las consultas RFC deben retornar datos consistentes con filtros, ordenamientos y agregaciones especificadas.