Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

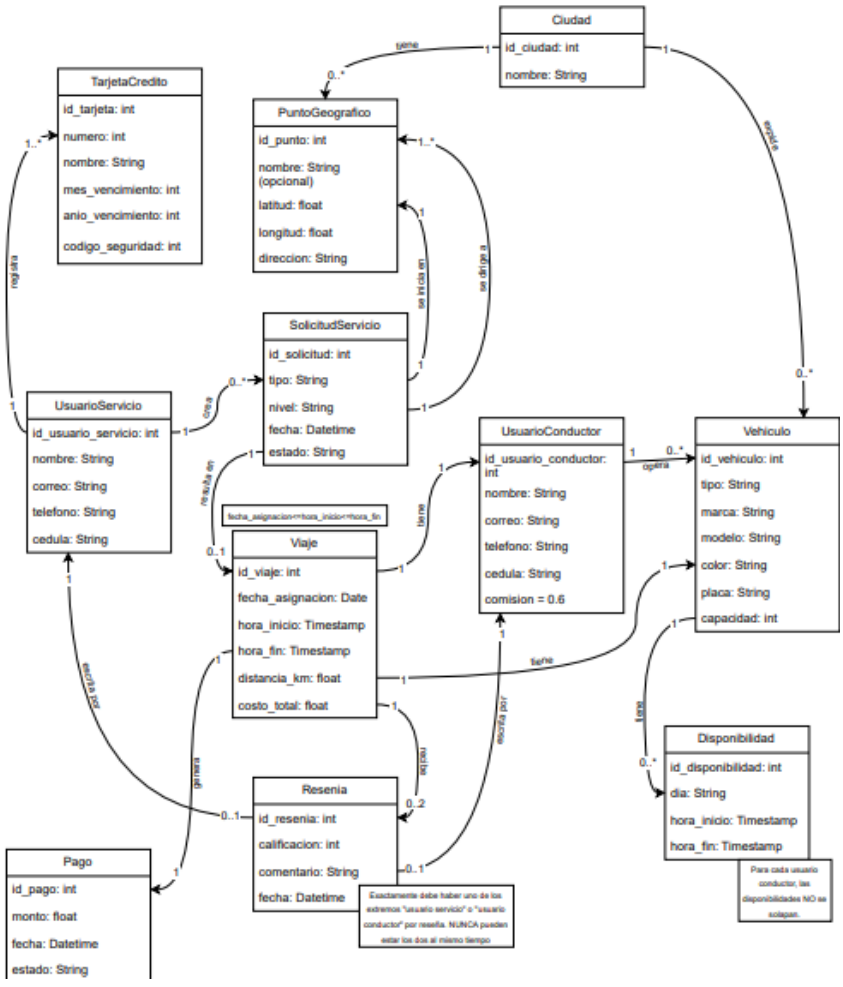
Proyecto AlpesCab

Entrega 2 - Implementación

Sistemas Transaccionales

1. Análisis y revisión del modelo de datos

1.1. Modelo conceptual UML



Clases principales: `Ciudad`, `PuntoGeografico`, `UsuarioServicio` (pasajero/cliente), `UsuarioConductor`, `Vehiculo`, `Disponibilidad`, `SolicitudServicio`, `Viaje`, `Pago`, `Resenia`, `TarjetaCredito`.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

Asociaciones clave y multiplicidades:

- `Ciudad` 1..* `PuntoGeografico`
- `UsuarioConductor` 1..* `Vehiculo`
- `Vehiculo` 1..* `Disponibilidad`
- `UsuarioServicio` 1..* `SolicitudServicio`
- `SolicitudServicio` 1..1 `Viaje`
- `Viaje` 1..1 `Pago`
- `Viaje` 1..* `Resenia` (una del cliente al conductor y otra del conductor alcliente)

Cambios realizados respecto a la iteración anterior:

- **Nivel del servicio**: se modela como atributo de `SolicitudServicio` (y se deriva de la capacidad/modelo del `Vehiculo` cuando aplica a pasajeros), en lugar de estar en `Vehiculo`. Esto alinea el dato con la solicitud concreta y evita persistirlo en el vehículo de forma rígida. (La especificación indica que para pasajeros "el modelo y la capacidad del vehículo determinan el nivel", por lo que el nivel es una característica de la prestación en un contexto concreto.)
- **Tarifa**: no se persiste una tabla de tarifas en esta iteración; el **costo total** se registra en `Viaje` después del servicio. La guía del caso menciona tarifas por kilómetro como criterio de cálculo; nosotros lo dejamos parametrizable/fuera **de alcance** en el modelo físico y calculado en lógica de negocio, guardando el resultado en `Viaje.costo_total`.
- **Disponibilidad**: se asocia a **conductor y vehículo** con checks para **no** solapamiento por día/horarios y tipos de servicio (pasajeros, comida, mercancías), como pide RF5–RF6.
- **Reseñas**: se mantienen como entidad separada vinculada a `Viaje`, permitiendo evaluación 0..5 y comentario, conforme a la descripción del negocio.
- **TarjetaCredito**: entidad asociada a `UsuarioServicio` para medios de pago.

1.2. Transformación UML → Relacional (algoritmo modificado de Chen)

A continuación, se **justifica** la selección de tablas y claves, usando las reglas vistas en clase:

- **CIUDAD (**id_ciudad PK, nombre UNIQUE): Clase fuerte → relación propia. Unique(nombre) para evitar duplicados por homónimos.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

- **PUNTO_GEOGRAFICO (**id_punto PK, nombre, latitud, longitud, direccion, id_ciudad FK): Clase fuerte con **FK a CIUDAD** (asociación 1..*).
- **USUARIO_SERVICIO (**id_usuario_servicio PK, nombre, correo, teléfono, cédula …): Clase fuerte → relación propia.
- **USUARIO_CONDUCTOR (**id_usuario_conductor PK, nombre, correo, teléfono, cédula, comisión …): Clase fuerte → relación propia.
- **VEHICULO (**id_vehiculo PK, tipo, marca, modelo, color, placa UNIQUE, capacidad, id_usuario_conductor FK, id_ciudad_expedicion FK): Clase fuerte; **asociación 1..*** de conductor a vehículo se implementa con **FK en VEHICULO** a `USUARIO_CONDUCTOR`.
- **DISPONIBILIDAD (**id_disponibilidad PK, dia, hora_inicio, hora_fin, tipo_servicio, id_vehiculo FK, id_usuario_conductor FK): Asociación de disponibilidad como entidad con atributos propios (intervalos y tipo). Se usan **checks** (ej. `hora_inicio < hora_fin`) y validaciones para evitar solapes por día. (Regla: asociación con atributos ⇒ relación propia con FKs a sus extremos.)
- **SOLICITUD_SERVICIO (**id_solicitud PK, tipo, nivel, fecha, estado, id_usuario_servicio FK, id_punto_partida FK): Clase fuerte que captura la intención del usuario.
- **VIAJE (**id_viaje PK, fecha_asignacion, hora_inicio, hora_fin, distancia_km, costo_total, id_usuario_conductor FK, id_vehiculo FK, id_punto_partida FK, id_solicitud FK UNIQUE): Representa la ejecución; se impone **1:1 con SOLICITUD** a través de `UNIQUE(id_solicitud)` (o verificación en servicio) para reflejar una asignación por solicitud en esta iteración.
- **PAGO (**id_pago PK, fecha, estado, monto, id_viaje FK UNIQUE): Un pago por viaje; `UNIQUE(id_viaje)` refuerza 1:1.
- **RESENIA (**id_resenia PK, calificacion 0..5, comentario, fecha, id_viaje FK, id_usuario_origen FK, id_usuario_destino FK): Permite una reseña del cliente al conductor y viceversa por viaje.
- **TARJETA_CREDITO (**id_tarjeta PK, numero UNIQUE, nombre, mes_vencimiento, anio_vencimiento, codigo_seguridad, id_usuario_servicio FK): Medio de pago del usuario de servicio.

Nota: El **detalle tabla por tabla** con dominios/PK/FK/UNIQUE/CK se documentó en la **plantilla Excel** de BN como exige la entrega (ver Anexo de Modelo Relacional).

Tablas de entidades

1. **Ciudad**

Ciudad

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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(250CHAR))	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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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(250CHAR))	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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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"

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)

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

NN, CK, UA	NN, UA	NN, FK$_{UsuarioServicio.id_usuario_servicio}$
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)	latitud (FLOAT) nombre (VARCHAR2(250CHAR))	longitud (FLOAT)
PK, NN, ND, SA	NN NN, UA	NN
1001	4.711 Oficina Norte	-740.721
1002	46.687 Centro Comercial Andino	-740.544
1003	62.088 Parque Lleras	-75.566
1004	47.016 Aeropuerto El Dorado	-741.469
1005	34.516 Terminal Cali	-765.319
direccion (VARCHAR2(250CHAR))		id_ciudad (NUMBER)
NN		NN, FK$_{Ciudad.id_ciudad}$
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).

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

						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.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

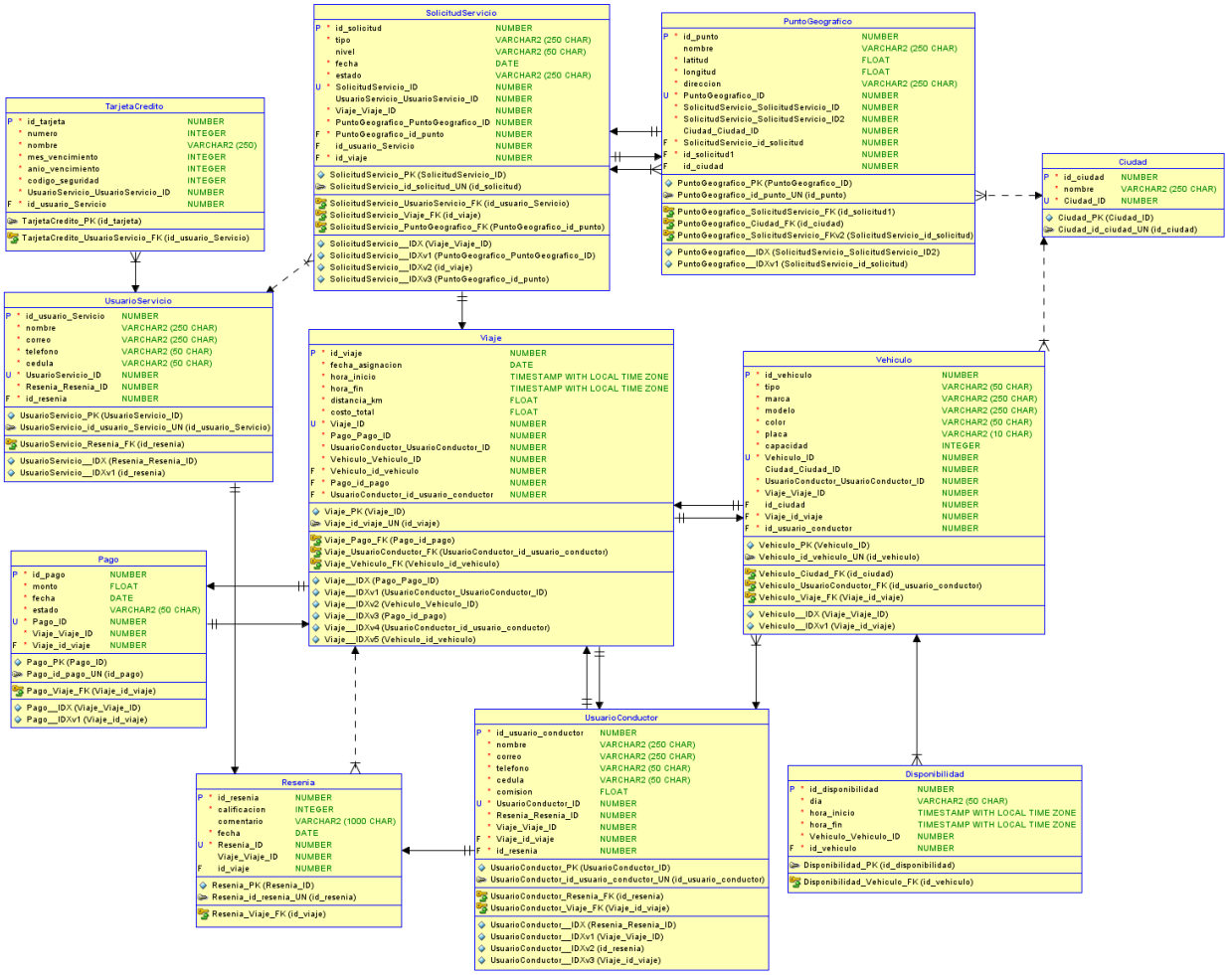
Disponibilidad	Asociación UML entre Conductor y Vehículo con atributos	R2 (Relación 1:N → FK en el lado N)	PK: id_disponibilidad	FKs: Vehiculo.id_vehiculo	rango horario válido	Se convierte en tabla propia porque tiene atributos adicionales (día, hora_inicio, hora_fin, tipo_servicio). No podía representarse 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 → FK en el lado N)	id_solicitud	UsuarioServicio, PuntoGeografico, Viaje	dominios de tipo, nivel, estado	Representa la orden generada por el cliente. Incluye FKs a UsuarioServicio y PuntoGeografico

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

						(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.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



1.3. Verificación de BCNF

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:

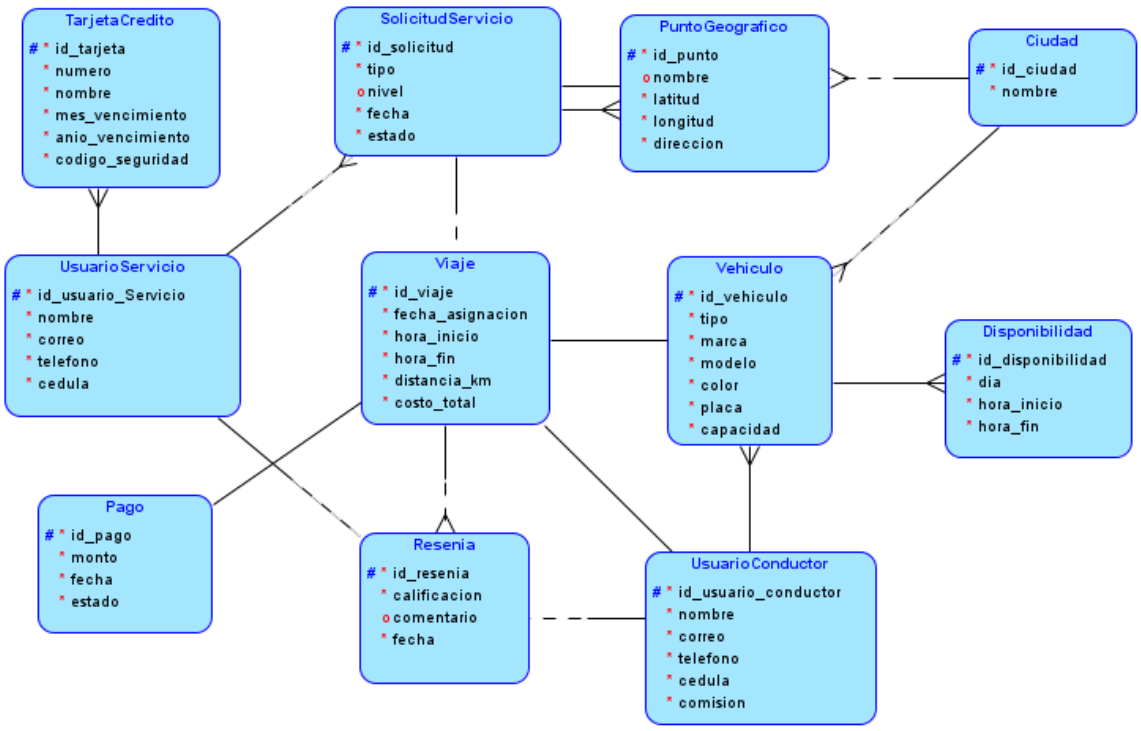
- El modelo cumple con la 3NF y todas las llaves son simples.

Se analizó cada relación identificando dependencias funcionales (DF) y claves candidatas:

- **CIUDAD**(id_ciudad → nombre). Clave: id_ciudad. `nombre` depende totalmente de la clave. **BCNF**.
- **PUNTO_GEOGRAFICO**(id_punto → nombre, latitud, longitud, direccion, id_ciudad). Clave: id_punto. **BCNF**.
- **USUARIO_SERVICIO / USUARIO_CONDUCTOR**: todos los atributos dependen de la **PK**; los identificadores alternos (correo, cédula) se controlan con **UNIQUE**. **BCNF**.
- **VEHICULO**: (id_vehiculo → tipo, marca, modelo, color, placa, capacidad, id_usuario_conductor, id_ciudad_expedicion). **BCNF**; `placa` se declara **UNIQUE**.
- **DISPONIBILIDAD**: (id_disponibilidad → dia, hora_inicio, hora_fin, tipo_servicio, id_vehiculo, id_usuario_conductor). Observación: en el dominio, **un vehículo pertenece a un solo conductor**; por tanto existe DF **id_vehiculo → id_usuario_conductor**. Como **id_vehiculo no es superclave en DISPONIBILIDAD**, mantener ambos atributos puede **violentar BCNF estricta**.
    - **Decisión**: se mantiene `id_usuario_conductor` por **eficiencia y reglas RF5–RF6** (validar solapes a nivel conductor y vehículo), con **FKs** y validaciones para garantizar consistencia → **desnormalización controlada**. Alternativa BCNF estricta: eliminar `id_usuario_conductor` y obtenerlo por *join* a `VEHICULO`.
- **SOLICITUD_SERVICIO**: (id_solicitud → tipo, nivel, fecha, estado, id_usuario_servicio, id_punto_partida). **BCNF**.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

- **VIAJE**: (id_viaje → fecha_asignacion, hora_inicio, hora_fin, distancia_km, costo_total, id_usuario_conductor, id_vehiculo, id_punto_partida, id_solicitud). **BCNF**; se controla 1:1 con `SOLICITUD` por restricción/validación.
- **PAGO**: (id_pago → fecha, estado, monto, id_viaje). **BCNF**; `id_viaje` con **UNIQUE** asegura 1:1.
- **RESENIA**: (id_resenia → calificacion, comentario, fecha, id_viaje, id_usuario_origen, id_usuario_destino). **BCNF**.
- **TARJETA_CREDITO**: (id_tarjeta → numero, nombre, mes_vencimiento, anio_vencimiento, codigo_seguridad, id_usuario_servicio). **BCNF**; `numero` **UNIQUE**.



Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

2. Arquitectura de la implementación del esquema en Oracle

2.1. Convenciones de modelado físico y estándares

- **Esquema**: `ISIS2304A08202520`. Objetos en **MAYÚSCULAS**.
- **Tablas**: nombre en singular; atributos con "_"
- **PK**: `PK_<TABLA>`; **FK**: `FK_<TABLA>_<REF>`; **UNIQUE**: `UQ_<TABLA>_<COL>`; **CHECK**: `CK_<TABLA>_<REGLA>`.
- **Tipos de datos**: `NUMBER` para identificadores y valores numéricos; `VARCHAR2(n CHAR)` para texto; `DATE/TIMESTAMP` según requerimiento.
- **Índices**: índice explícito en cada **FK** y en columnas **UNIQUE** de búsqueda frecuente (ej. `placa`).
- **Autonumeración**: cuando un ID puede no venir desde la aplicación, se define SEQUENCE + TRIGGER BEFORE INSERT.
- **Integridad**: NOT NULL, UNIQUE y CHECK en la BD (no solo a nivel aplicación).

2.2. Tablas y columnas principales

- **CIUDAD (**id_ciudad PK, nombre UNIQUE)
- **PUNTO_GEOGRAFICO (**id_punto PK, nombre, latitud, longitud, direccion, id_ciudad FK)
- **USUARIO_SERVICIO (**id_usuario_servicio PK, nombre, correo, telefono, cedula …)
- **USUARIO_CONDUCTOR (**id_usuario_conductor PK, nombre, correo, telefono, cedula, comision …)
- **VEHICULO (**id_vehiculo PK, tipo, marca, modelo, color, placa UNIQUE, capacidad, id_usuario_conductor FK, id_ciudad_expedicion FK)
- **DISPONIBILIDAD (**id_disponibilidad PK, dia, hora_inicio, hora_fin, tipo_servicio, id_vehiculo FK, id_usuario_conductor FK)
- **SOLICITUD_SERVICIO (**id_solicitud PK, tipo, nivel, fecha, estado, id_usuario_servicio FK, id_punto_partida FK)
- **VIAJE (**id_viaje PK, fecha_asignacion, hora_inicio, hora_fin, distancia_km, costo_total, id_usuario_conductor FK, id_vehiculo FK, id_punto_partida FK, id_solicitud FK UNIQUE)
- **PAGO (**id_pago PK, monto, fecha, estado, id_viaje FK UNIQUE)

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

- **RESENIA (**id_resenia PK, calificacion, comentario, fecha, id_viaje FK [, id_usuario_origen FK, id_usuario_destino FK si aplica])
- **TARJETA_CREDITO (**id_tarjeta PK, numero UNIQUE, nombre, mes_vencimiento, anio_vencimiento, codigo_seguridad, id_usuario_servicio FK)

3. Arquitectura de la implementación de la lógica en la aplicación

3.1. Arquitectura y capas

- **Backend**: Spring Boot (Java).
- **Capa de exposición**: controladores `@RestController` con rutas HTTP que siguen el enunciado (RF1–RF11 y RFC1–RFC4).
- **Capa de acceso a datos**: repositorios Spring Data JPA con **consultas nativas** (INSERT/UPDATE/SELECT) y proyecciones para reportes.
- **Base de datos**: Oracle. Integridad garantizada por **PK, FK, UNIQUE y CHECK**; algunas reglas también se validan en la capa de servicio para devolver errores de negocio claros (400/404/409).

Req	Método	Ruta	Entrada (params/body)	Salida (éxito)	Reglas/validaciones clave
RF1	POST	`/ciudades/new/save`	JSON: `{"nombre"}`	`200 OK {idCiudad, nombre}`	Unique(nombre). Autonumeración (SEQ+TRIGGER). 409 si ya existe. 400 si nombre vacío.
RF2	POST	`/usuarios-servicio/new/save`	JSON: `{id, nombre, correo, telefono, cedula}`	`200 OK`	PK nueva. Campos obligatorios. 409 si id/correo/cédula existen (según restricciones).
RF3	POST	`/usuarios-conductor/new/save`	JSON: `{id, nombre, correo, telefono, cedula, comision }`	`200 OK`	PK nueva. `comision` en [0,1]. 409 si duplicados.
RF4	POST	`/vehiculos/registrar`	form: `idVehiculo, tipo, marca, modelo, color, placa, capacidad,`	`200 OK`	409 si `idVehiculo` o `placa` existen. 404 si conductor/ciudad no existen.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

			idUsuarioConductor, idCiudadExpedicion		
RF5	POST	/disponibilidades/registrar	form: idDisponibilidad, dia, horaInicio, horaFin, tipoServicio, idVehiculo, idUsuarioConductor	200 OK	Formato fecha YYYY-MM-DD HH:MM:SS. hora_fin > hora_inicio. Día y tipo en listas permitidas. 404 si FK no existen. **409 si hay traslape** (mismo día con vehículo o conductor).
RF6	POST	/disponibilidades/registrar	Igual a RF5 (otro tipo/horario)	200 OK	Mismas validaciones que RF5.
RF7	POST	/puntos/new/save	JSON: { idPunto, nombre, latitud, longitud, direccion, idCiudad }	200 OK	404 si ciudad no existe. Checks de lat/long si aplica.
RF8a	POST	/solicitudes/registrar	form: idSolicitud, tipo, nivel, fecha (YYYY-MM-DD), estado, idUsuarioServicio, idPuntoPartida	200 OK	404 si usuario/punto no existen. fecha válida.
RF8b	POST	/viajes/registrar	form: idViaje, fechaAsignacion (YYYY-MM-DD), horaInicio (opt), horaFin (opt), distanciaKm (opt), costoTotal (opt), idUsuarioConductor, idVehiculo, idPuntoPartida, idSolicitud	200 OK	404 si alguna FK no existe. **1:1**: 409 si la solicitud ya tiene viaje. Si BD exige NOT NULL en distancia/costo, se envían 0.0 por defecto. Validación temporal si vienen ambas horas.
RF9	POST	/viajes/finalizar	form: idViaje, horaInicio (opt), horaFin, distanciaKm, costoTotal	200 OK	404 si viaje no existe. Si hora_inicio estaba NULL en BD, se **exige** horaInicio. horaFin > horaInicio. Distancia/costo válidos.
RF10	POST	/resenias/registrar	form: idResenia, idViaje, calificacion,	200 OK	404 si viaje no existe. calificacion ∈ [0..5].

			comentario, fecha (YYYY-MM-DD)		
RF11	POST	/resenias/registrar	form: idResenia, idViaje, calificacion, comentario, fecha (YYYY-MM-DD)	200 OK	Puede usarse para conductor→cliente. Validaciones equivalentes.
RFC1	GET	/reportes/historico/usuarios-servicio/{id}?fini&ffin	Path+query	200 OK [{idSolicitud, tipo, nivel, fecha, estado, fechaAsignacion, horaInicio, horaFin, distanciaKm, costoTotal}]	Filtros de fecha opcionales. Join solicitud_servicio + viaje.
RFC2	GET	/reportes/top-conductores?fini&ffin	Query	200 OK [{idConductor, nombre, totalServicios}]	Rango por hora_inicio. Orden desc, top 20.
RFC3	GET	/reportes/ganancias-conductor/{id}?fini&ffin	Path+query	200 OK [{idVehiculo, placa, tipoServicio, ganadoConductor}]	Suma (costo_total * comision) agrupado por vehículo y tipo de servicio.
RFC4	GET	/reportes/utilizacion?idCiudad&fini&ffin	Query	200 OK [{tipo, nivel, totalServicios, porcentaje}]	Cuenta por tipo/nivel de **solicitudes** en rango y ciudad (punto partida). Orden por uso desc.

3.3. Reglas de negocio y validaciones implementadas

- **Unicidad**: CIUDAD.nombre, VEHICULO.placa, IDs de todas las entidades. Respuesta **409** si se viola.
- **Presencia de FK**: antes de INSERT/UPDATE se verifica la existencia de la entidad referenciada (ciudad, usuario, vehículo, punto…). Si falta: **404**.
- **Disponibilidad**:
  - dia ∈ {LUNES, …, DOMINGO}.
  - tipo_servicio ∈ {PASAJEROS, COMIDA, MERCANCÍAS/MERCANCIAS}.
  - hora_fin > hora_inicio y **no solape** por (día, vehículo) **o** (día, conductor). Si hay solape: **409**.

- **Solicitud → Viaje (1:1)**: una solicitud no puede tener más de un viaje. Intento de duplicar: **409**.
- **Tiempos de viaje**: se valida formato (`YYYY-MM-DD` y `YYYY-MM-DD HH:MM:SS`) y coherencia `hora_fin > hora_inicio` cuando ambas están presentes.
- **Finalización de viaje**: si el viaje comenzó sin `hora_inicio` (NULL en BD), se **exige** que se envíe `horaInicio` al finalizar; de lo contrario **400**.
- **Valores numéricos**: `distanciaKm` y `costoTotal` no negativos. En registro de viaje se admiten en `0` si la tabla los marca NOT NULL.
- **Reseñas**: `calificacion` en [0..5]; comentario opcional; ligada a un `id_viaje` válido.

3.4. Manejo de errores

- **200 OK**: operación exitosa (crear/actualizar/consultar).
- **400 Bad Request**: formatos inválidos (fechas/horas), reglas de negocio incumplidas (p.ej., `hora_fin ≤ hora_inicio`, día/tipo no permitido).
- **404 Not Found**: FK inexistente o recurso base no encontrado (p.ej., `vehículo`, `punto`, `viaje`).
- **409 Conflict**: violación de integridad o reglas de unicidad/1:1/solapes.
- **500 Internal Server Error**: error no controlado (se minimiza validando antes de ejecutar SQL).

La aplicación incluye un manejador global de excepciones para retornar errores **409** con detalle cuando la BD dispara una restricción.

3.5. Evidencia con Postman

Se entrega una **colección Postman** con todos los RF/RFC y ejemplos de ejecución (casos exitosos y fallas controladas). Los requests usan `JSON` y `x-www-form-urlencoded` donde se definieron `@RequestParam`. Para la evidencia, se incluyen capturas de:

Éxitos (200)

- ```
  # RF1 — Crear ciudad (JSON)
  curl -X POST "http://localhost:8080/ciudades/new/save" \
    -H "Content-Type: application/json" \
  ```
  ```

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

```
  -d '{"nombre": "Evid_Ciudad_01"}'
```

- ```
  # RF2 — Crear UsuarioServicio (JSON)
  curl -X POST "http://localhost:8080/usuarios-servicio/new/save" \
    -H "Content-Type: application/json" \
    -d '{"id": 11001, "nombre": "Eva Servicio", "correo":
  "eva.servicio@demo.com", "telefono": "3000000001", "cedula": "CC-
  EV001"}'
  ```

- ```
  # RF3 — Crear UsuarioConductor (JSON)
  curl -X POST "http://localhost:8080/usuarios-conductor/new/save"
  \
    -H "Content-Type: application/json" \
    -d '{"id": 12001, "nombre": "Carlos Conductor", "correo":
  "carlos.conductor@demo.com", "telefono": "3000000002", "cedula":
  "CC-CD001", "comision": 0.6}'
  ```

- ```
  # RF7 — Crear Punto geográfico (JSON)
  curl -X POST "http://localhost:8080/puntos/new/save" \
    -H "Content-Type: application/json" \
    -d '{"idPunto": 14001, "nombre": "Punto Evidencia", "latitud":
  4.70, "longitud": -74.10, "direccion": "CR 1 # 2-3", "idCiudad":
  100}'
  ```

- ```
  # RF4 — Registrar Vehículo (form)
  curl -X POST "http://localhost:8080/vehiculos/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d
  "idVehiculo=13001&tipo=CARRO&marca=Toyota&modelo=Corolla&color=Ro
  jo&placa=EVX001&capacidad=4&idUsuarioConductor=12001&idCiudadExpe
  dicion=100"
  ```

- ```
  # RF5 — Registrar Disponibilidad (form, sin solape)
  curl -X POST "http://localhost:8080/disponibilidades/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idDisponibilidad=18001&dia=LUNES&horaInicio=2025-09-01
  08:00:00&horaFin=2025-09-01
  12:00:00&tipoServicio=PASAJEROS&idVehiculo=13001&idUsuarioConduct
  ```
  ```

```
or=12001"
```

- ```
  # RF8a — Crear Solicitud de Servicio (form)
  curl -X POST "http://localhost:8080/solicitudes/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idSolicitud=15001&tipo=PASAJEROS&nivel=ESTANDAR&fecha=2025-
  09-01&estado=CREADA&idUsuarioServicio=11001&idPuntoPartida=14001"
  ```

- ```
  # RF8b — Registrar Viaje (form)
  curl -X POST "http://localhost:8080/viajes/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idViaje=16001&fechaAsignacion=2025-09-01&horaInicio=2025-
  09-01
  14:00:00&distanciaKm=0&costoTotal=0&idUsuarioConductor=12001&idVe
  hiculo=13001&idPuntoPartida=14001&idSolicitud=15001"
  ```

- ```
  # RF9 — Finalizar Viaje (form)
  curl -X POST "http://localhost:8080/viajes/finalizar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idViaje=16001&horaInicio=2025-09-01 14:00:00&horaFin=2025-
  09-01 14:35:00&distanciaKm=10.5&costoTotal=25000"
  ```

- ```
  # RF10 — Reseña Cliente→Conductor (form)
  curl -X POST "http://localhost:8080/resenias/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d
  "idResenia=19001&idViaje=16001&calificacion=5&comentario=Excelent
  e servicio&fecha=2025-09-01"
  ```

- ```
  # RF11 — Reseña Conductor→Cliente (form)
  curl -X POST "http://localhost:8080/resenias/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d
  "idResenia=19002&idViaje=16001&calificacion=4&comentario=Pasajero
  puntual&fecha=2025-09-01"
  ```
  ```

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

Unicidad (409)

- ```
  # RF1 — Duplicar nombre de ciudad
  curl -X POST "http://localhost:8080/ciudades/new/save" \
    -H "Content-Type: application/json" \
    -d '{"nombre": "Evid_Ciudad_01"}'
  ```

- ```
  # RF4 — Duplicar placa
  curl -X POST "http://localhost:8080/vehiculos/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d
  "idVehiculo=13002&tipo=CARRO&marca=Toyota&modelo=Yaris&color=Azul
  &placa=EVX001&capacidad=4&idUsuarioConductor=12001&idCiudadExpedi
  cion=100"
  ```

- ```
  # RF5 — Solape de disponibilidad
  curl -X POST "http://localhost:8080/disponibilidades/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idDisponibilidad=18002&dia=LUNES&horaInicio=2025-09-01
  10:00:00&horaFin=2025-09-01
  11:00:00&tipoServicio=PASAJEROS&idVehiculo=13001&idUsuarioConduct
  or=12001"
  ```

- ```
  # RF8b — Asignar segundo viaje a la misma solicitud
  curl -X POST "http://localhost:8080/viajes/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idViaje=16002&fechaAsignacion=2025-09-01&horaInicio=2025-
  09-01
  15:00:00&distanciaKm=0&costoTotal=0&idUsuarioConductor=12001&idVe
  hiculo=13001&idPuntoPartida=14001&idSolicitud=15001"
  ```

FK inexistentes (404)

- ```
  # RF4 — Vehículo con ciudad inexistente
  curl -X POST "http://localhost:8080/vehiculos/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d
  "idVehiculo=13003&tipo=CARRO&marca=Kia&modelo=Rio&color=Blanco&pl
  ```
  ```

```
aca=EVX003&capacidad=4&idUsuarioConductor=12001&idCiudadExpedicio
n=999999"
```

- ```
  # RF7 — Punto con ciudad inexistente
  curl -X POST "http://localhost:8080/puntos/new/save" \
    -H "Content-Type: application/json" \
    -d '{"idPunto": 14002, "nombre": "Punto Error", "latitud":
  4.70, "longitud": -74.10, "direccion": "CR 4 # 5-6", "idCiudad":
  999999}'
  ```

- ```
  # RF8b — Viaje con solicitud inexistente
  curl -X POST "http://localhost:8080/viajes/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idViaje=16003&fechaAsignacion=2025-09-01&horaInicio=2025-
  09-01
  16:00:00&distanciaKm=0&costoTotal=0&idUsuarioConductor=12001&idVe
  hiculo=13001&idPuntoPartida=14001&idSolicitud=999999"
  ```

Validación (400)

- ```
  # RF5 — hora_fin <= hora_inicio
  curl -X POST "http://localhost:8080/disponibilidades/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idDisponibilidad=18003&dia=MARTES&horaInicio=2025-09-02
  10:00:00&horaFin=2025-09-02
  09:00:00&tipoServicio=PASAJEROS&idVehiculo=13001&idUsuarioConduct
  or=12001"
  ```

- ```
  # RF5 — día/tipo fuera de dominio
  curl -X POST "http://localhost:8080/disponibilidades/registrar" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -d "idDisponibilidad=18004&dia=FUNDAY&horaInicio=2025-09-03
  08:00:00&horaFin=2025-09-03
  12:00:00&tipoServicio=EXPRESS&idVehiculo=13001&idUsuarioConductor
  =12001"
  ```

- ```
  # RF9 — Finalizar viaje con horaFin < horaInicio
  curl -X POST "http://localhost:8080/viajes/finalizar" \
  ```
  ```

```
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "idViaje=16001&horaInicio=2025-09-01 14:00:00&horaFin=2025-
09-01 13:59:00&distanciaKm=10&costoTotal=20000"
```

Reportes (200)

- # RFC1 — Histórico por UsuarioServicio
  curl -X GET "http://localhost:8080/reportes/historico/usuarios-servicio/11001?fini=2025-08-01&ffin=2025-09-30"

- # RFC2 — Top 20 Conductores
  curl -X GET "http://localhost:8080/reportes/top-conductores?fini=2025-08-01&ffin=2025-09-30"

- # RFC3 — Ganancias por Conductor
  curl -X GET "http://localhost:8080/reportes/ganancias-conductor/12001?fini=2025-08-01&ffin=2025-09-30"

- # RFC4 — Utilización por Ciudad y Fechas
  curl -X GET "http://localhost:8080/reportes/utilizacion?idCiudad=100&fini=2025-08-01&ffin=2025-09-30"

4. Carga de datos y escenarios de prueba

4.1. Escenarios de prueba (cobertura por requerimiento)

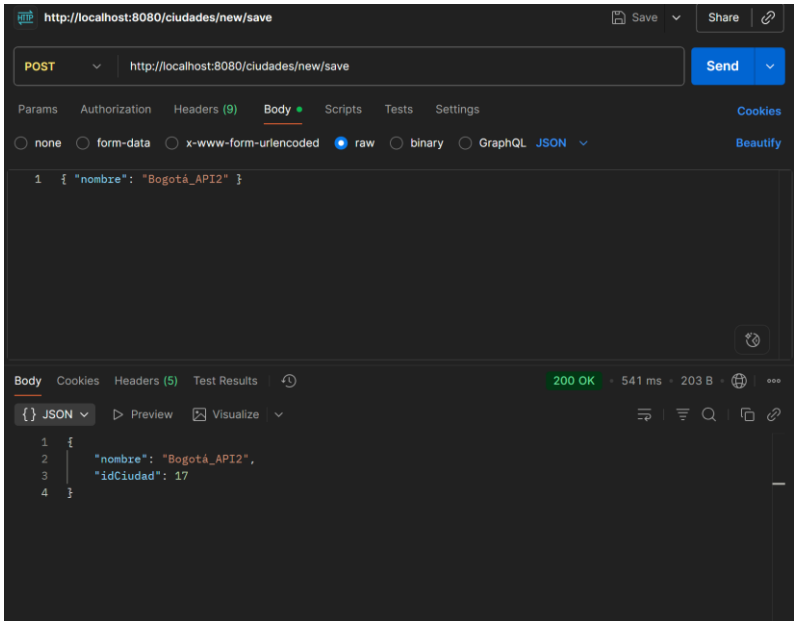
Cada RF con un caso exitoso y un caso de falla.

RF1 Registrar ciudad

Caso	Objetivo	Rol	Pasos	Resultado esperado

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

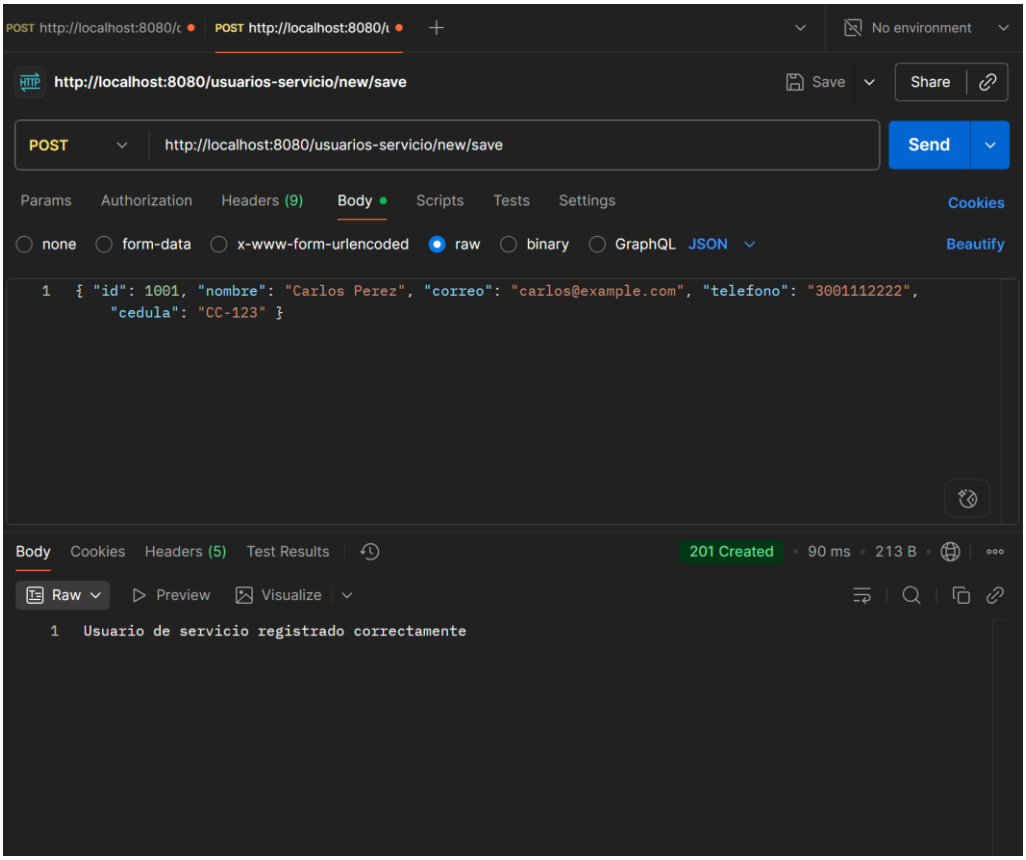
RF1 OK	Crear ciudad válida	ADM	Ingresar nombre “Barranquilla” y guardar	Ciudad creada, nombre único visible en catálogo
RF1 FAIL	Duplicado de nombre	ADM	Intentar crear “Bogotá” nuevamente	Rechazo por ND o mensaje de duplicado



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

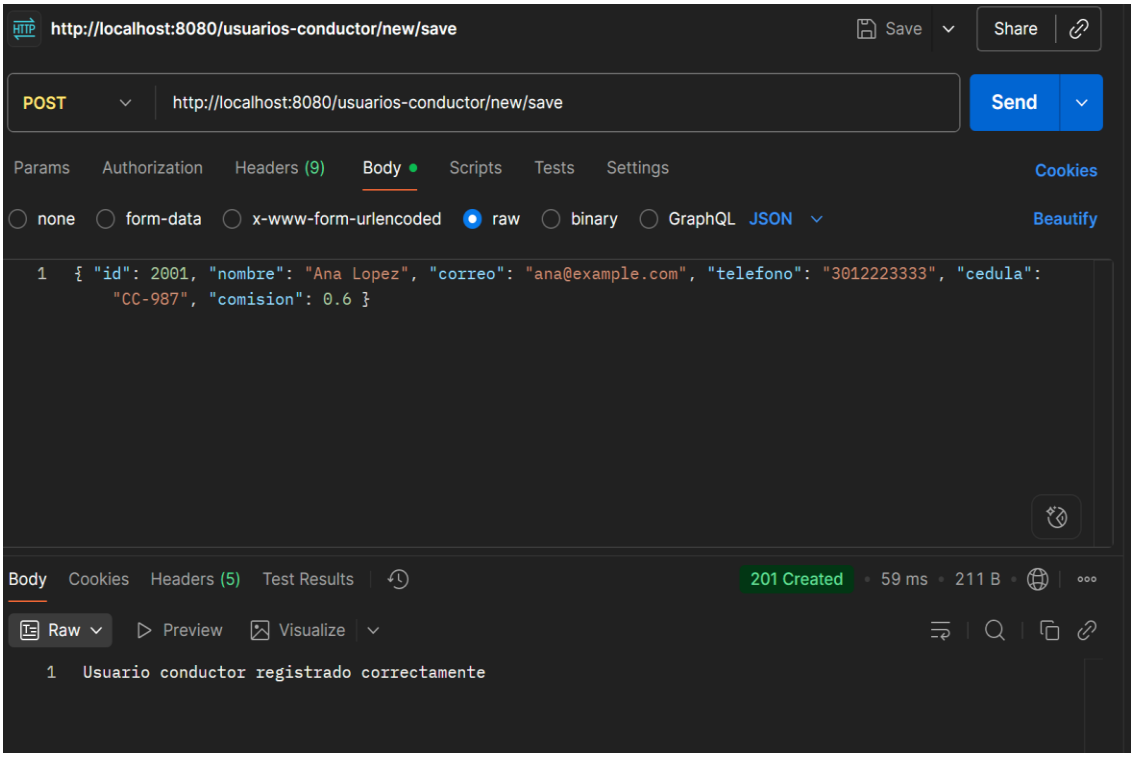
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



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

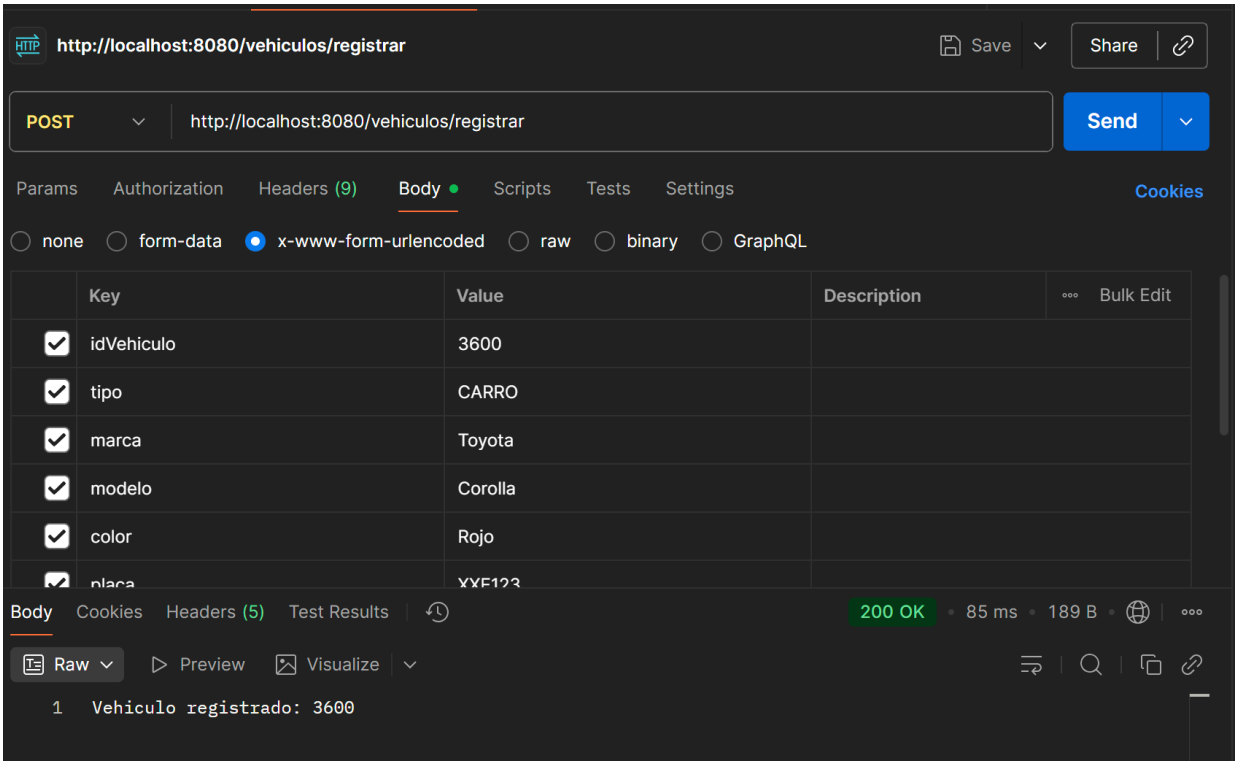
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



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

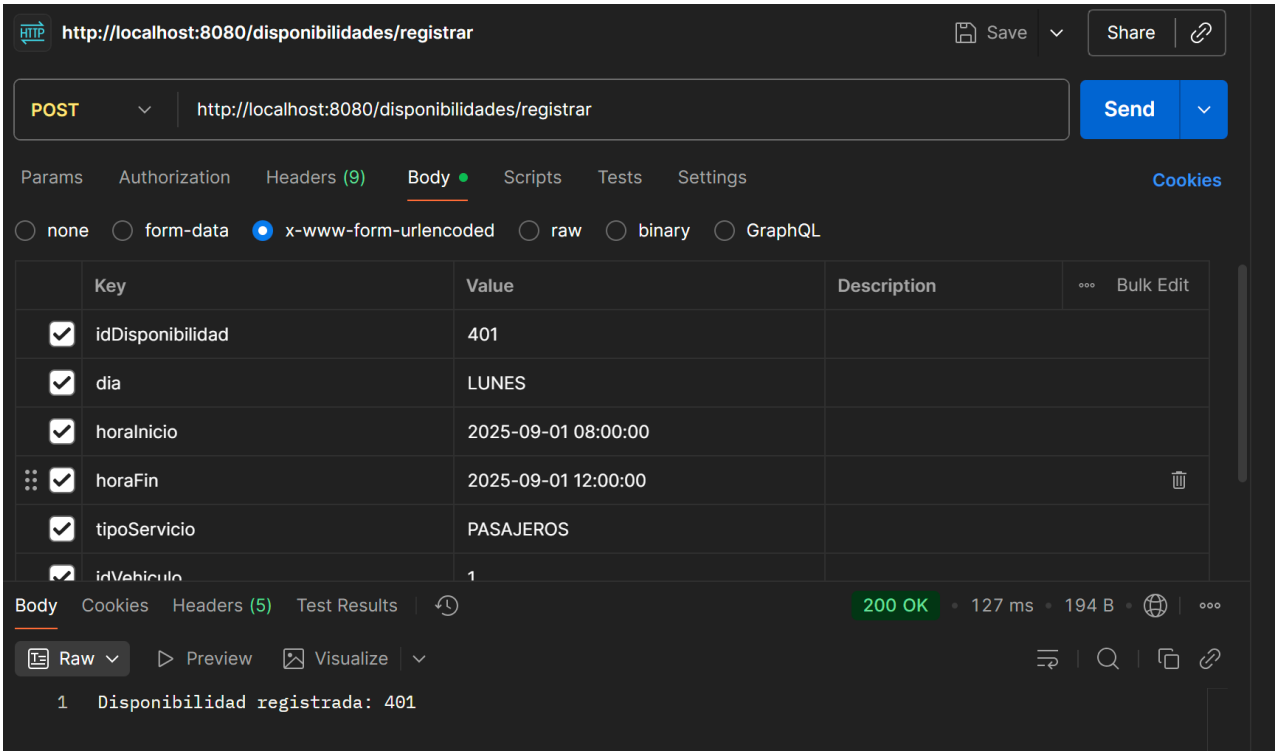
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



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

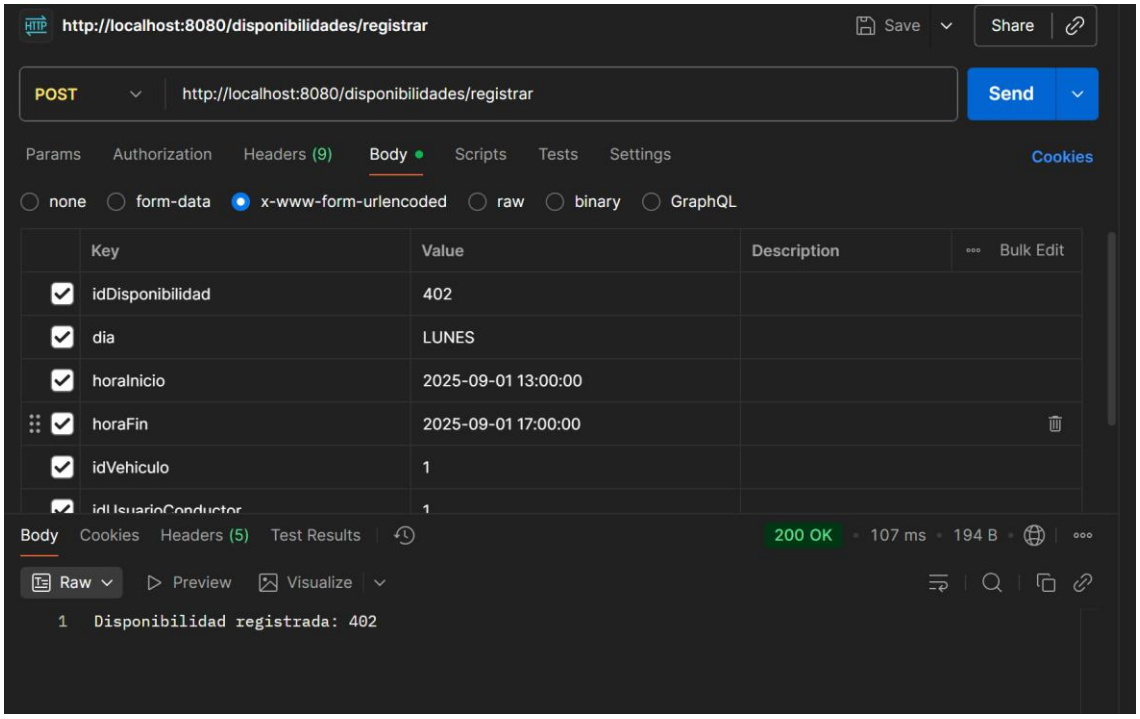
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



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

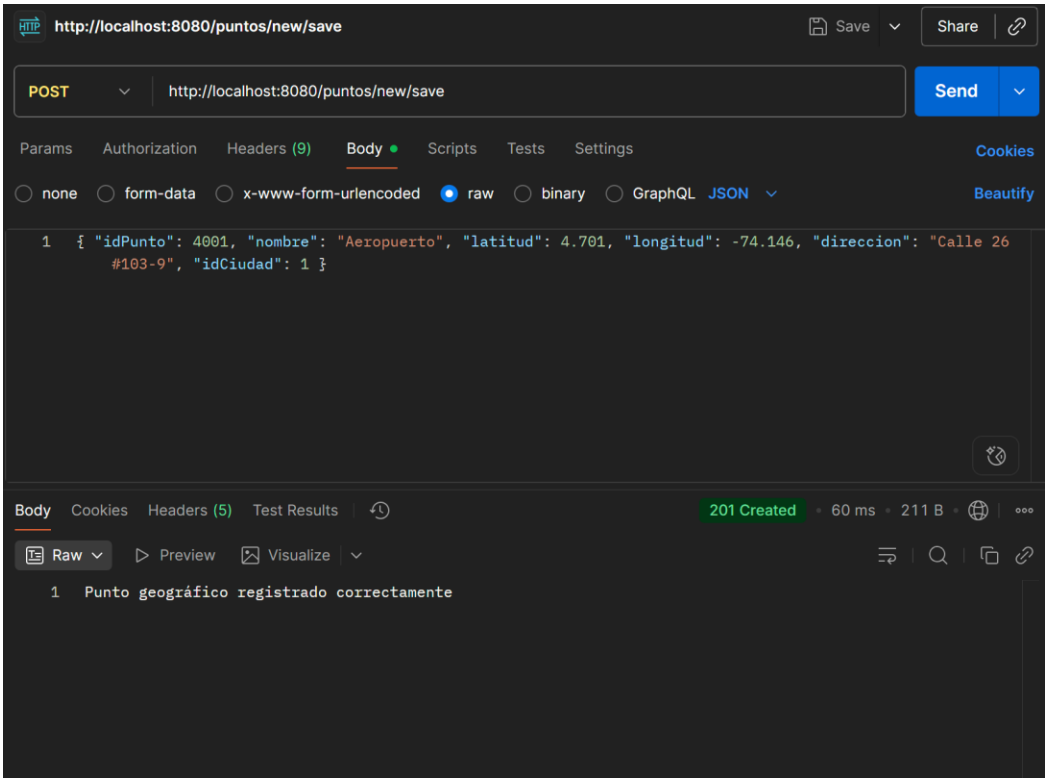
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

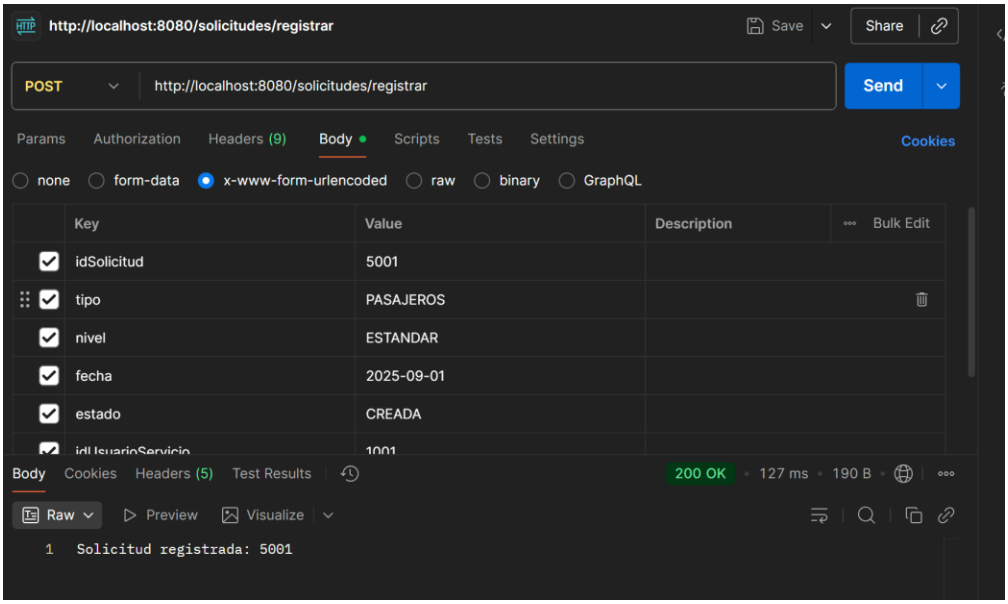


RF8 Solicitar servicio y asignación

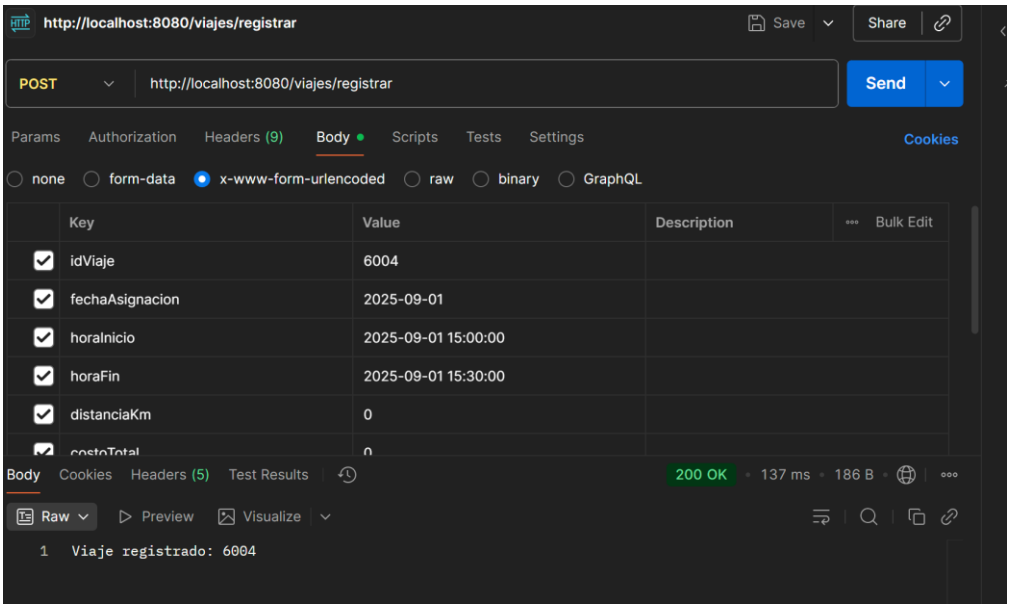
Caso	Objetivo	Rol	Pasos	Resultado esperado
RF8 OK	Crear solicitud válida y asignar	USR	Tipo PASAJEROS nivel ESTANDAR, origen 4001. App asigna conductor cercano disponible 2001 con 3001	Solicitud en estado ASIGNADA y Viaje creado con fecha_asignacion
RF8 FAIL	Sin disponibilidad	USR	Mismo origen y hora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

1. Crear Solicitud de Servicio

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



2. Asignar Viaje (registrar viaje)

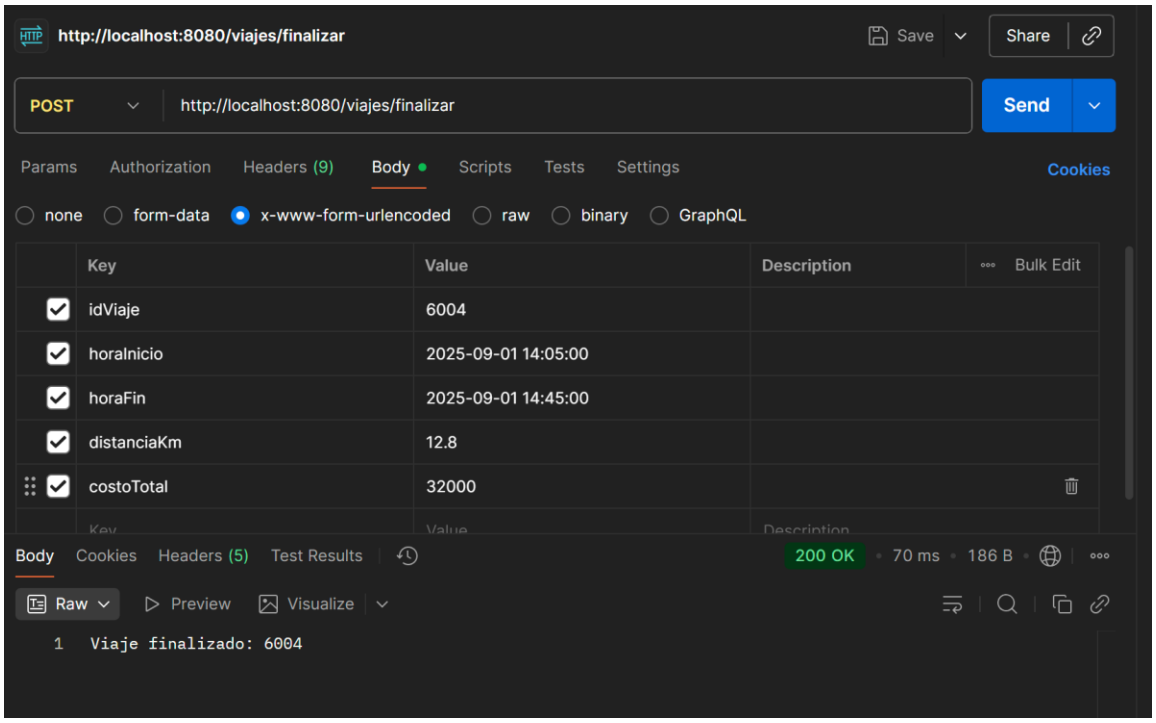


RF9 Registrar viaje terminado

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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

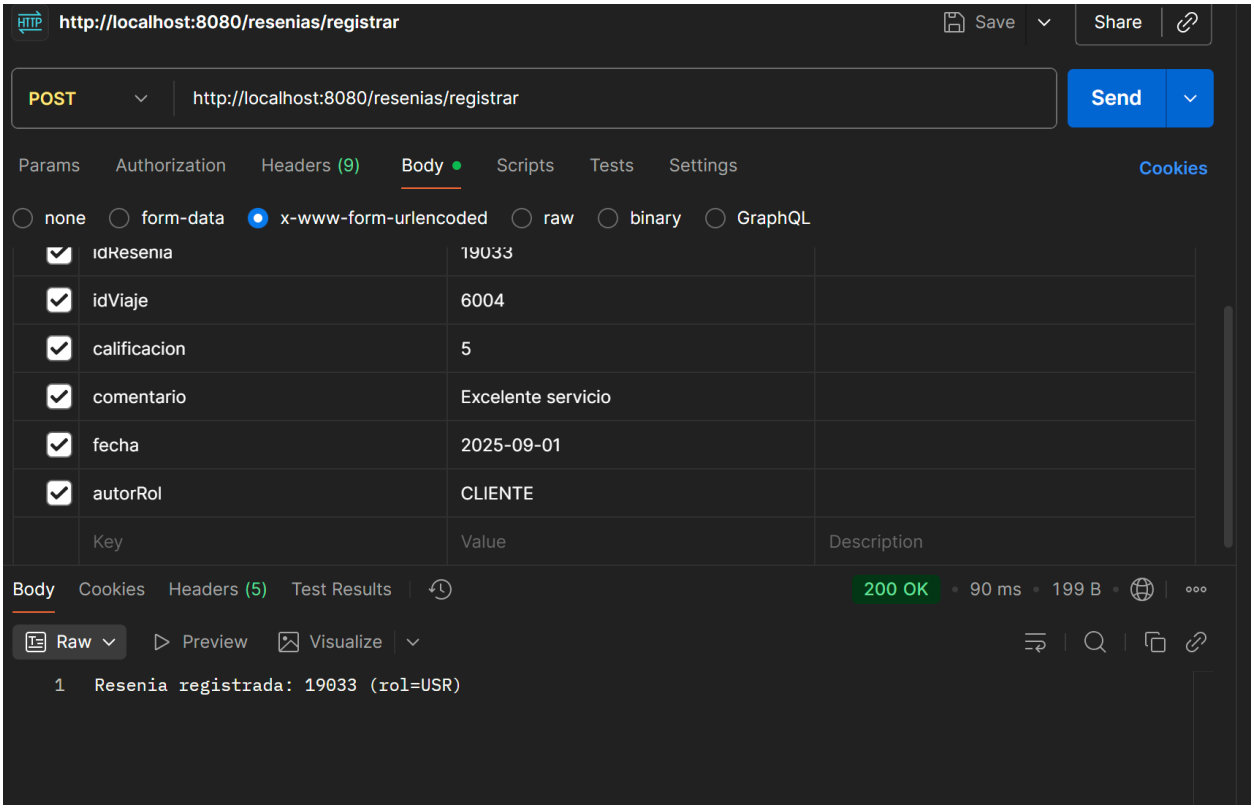
RF9 FAIL	Tiempos incoherentes	COND	Fin menor que inicio	Rechazo por check tiempos



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

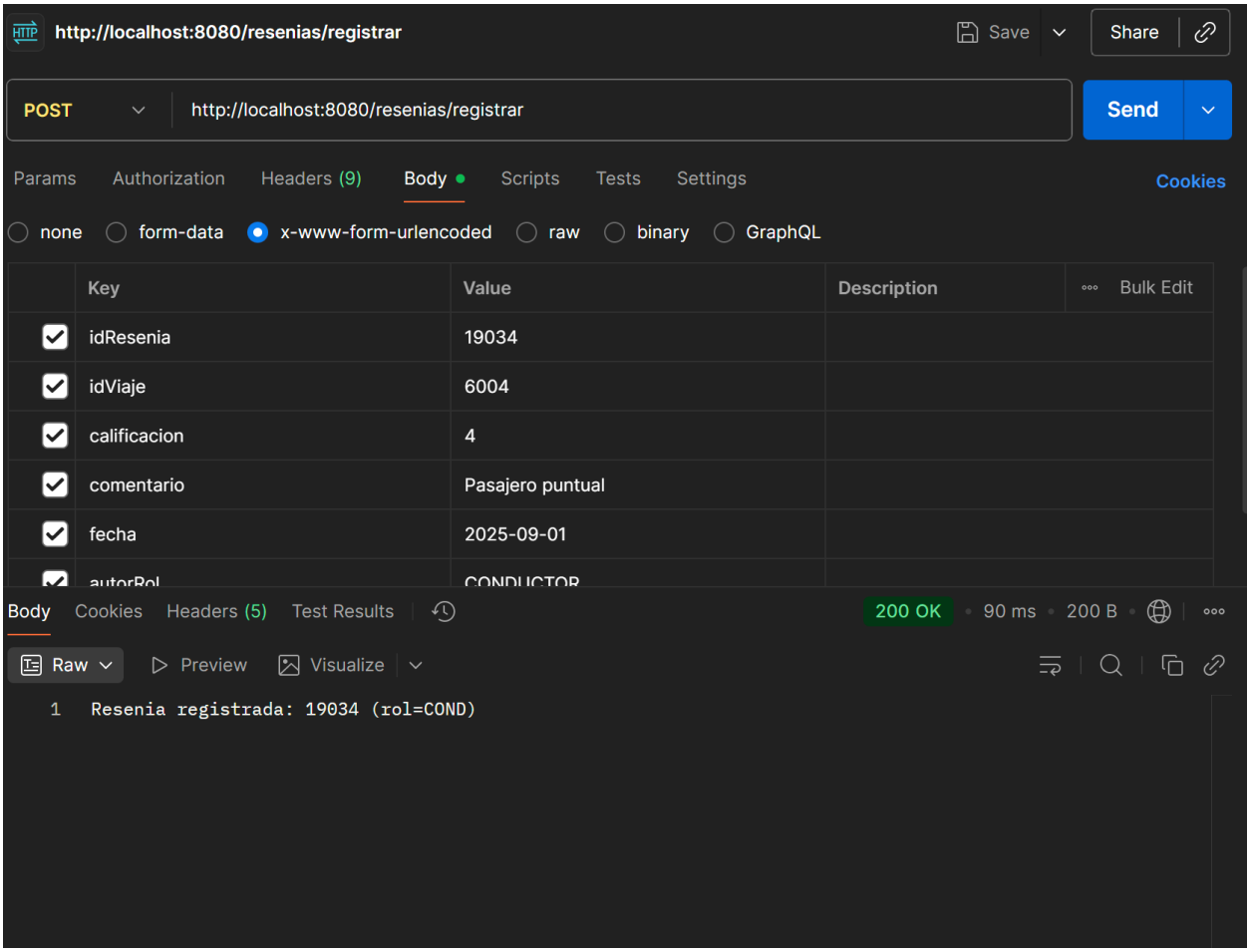
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



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

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



4.1.2 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
RFC1 A	Histórico UsuarioServicio	Usuario 1001 con	Ejecutar consulta de	Lista de solicitudes con estado y viajes asociados

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

		solicitudes y viajes	histórico para 1001	ordenados por fecha descendiente
RFC1 B	Histórico UsuarioConductor	Conductor 2001 con viajes	Ejecutar consulta de histórico para 2001	Lista de viajes con fechas y costos ordenados por fecha

La consulta tiene como objetivo mostrar el histórico de todos los servicios solicitados por un usuario específico, incluyendo la información de la solicitud, el viaje, el pago y la reseña asociada. Se usan las siguientes tablas: **USUARIO_SERVICIO**, que guarda los datos del pasajero (id, nombre, correo, teléfono, cédula); **SOLICITUD_SERVICIO**, donde se registran las solicitudes de servicio con sus atributos principales (id, tipo, nivel, estado, id del usuario que solicita); **VIAJE**, que representa la ejecución de la solicitud con datos de fechas, distancia, costo, id del conductor, id del vehículo y la relación con la solicitud; **VEHICULO**, con los datos del automóvil asignado (id, tipo, marca, modelo, color, placa, capacidad, id del conductor y ciudad de expedición); **PAGO**, que guarda los pagos de cada viaje (id, monto, fecha, estado e id del viaje); y **RESENIA**, que contiene las evaluaciones hechas después de un viaje (id, calificación de 0 a 5, comentario, fecha, autor rol e id del viaje). En cuanto a los joins, se usa un **INNER JOIN** entre USUARIO_SERVICIO y SOLICITUD_SERVICIO porque toda solicitud está asociada a un usuario. Los demás enlaces con VIAJE, CONDUCTOR, VEHICULO, PAGO y RESENIA se hacen con **LEFT JOIN** ya que puede haber solicitudes sin viaje, viajes sin conductor, viajes sin pago o sin reseña. Este esquema garantiza que siempre se muestren todos los servicios del usuario, con la información disponible en cada caso.


Resultado de la Consulta

Todas las Filas Recuperadas: 10 en 0,081 segundos

	ID_PASAJERO	NOMBRE_PASAJERO	CORREO_PASAJERO	TELEFONO_PASAJERO	ID_SOLICITUD	TIPO_SERVICIO	NIVEL_SERVICIO	ESTADO_SOLICITUD	ID_VIAJE	FECHA_ASIGNACION	HORA_INICIO	HORA_FIN	DISTANCIA_KM	COSTO_TOTAL	ID
1	1	Pasajero_1	pasajero1@alpescab.com	3005201472	300	PASAJEROS	ESTANDAR	CREADA	300	27/09/24	27/09/24	27/09/24	14,36	20043	
2	1	Pasajero_1	pasajero1@alpescab.com	3005201472	704	PASAJEROS	LARGE	CREADA	704	06/09/24	06/09/24	06/09/24	9,96	13579	
3	1	Pasajero_1	pasajero1@alpescab.com	3005201472	468	MERCANCIAS	LARGE	CREADA	468	15/06/24	15/06/24	15/06/24	7,12	11833	
4	1	Pasajero_1	pasajero1@alpescab.com	3005201472	538	PASAJEROS	ESTANDAR	CREADA	538	02/06/24	02/06/24	02/06/24	9,77	13601	
5	1	Pasajero_1	pasajero1@alpescab.com	3005201472	228	COMIDA	CONFORT	CREADA	228	25/03/24	25/03/24	25/03/24	13,08	16784	
6	1	Pasajero_1	pasajero1@alpescab.com	3005201472	633	PASAJEROS	CONFORT	CREADA	633	12/03/24	12/03/24	12/03/24	3,3	5894	
7	1	Pasajero_1	pasajero1@alpescab.com	3005201472	466	PASAJEROS	LARGE	CREADA	466	26/02/24	26/02/24	26/02/24	5,89	10253	
8	1	Pasajero_1	pasajero1@alpescab.com	3005201472	762	PASAJEROS	ESTANDAR	ASIGNADA	762	20/01/24	20/01/24	20/01/24	5,66	8303	
9	1	Pasajero_1	pasajero1@alpescab.com	3005201472	713	MERCANCIAS	LARGE	CREADA	713	11/01/24	11/01/24	11/01/24	4,6	7474	
10	1	Pasajero_1	pasajero1@alpescab.com	3005201472	149	PASAJEROS	LARGE	CREADA	149	08/01/24	08/01/24	08/01/24	3,04	7022	

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

Resultado de la Consulta

Todas las Filas Recuperadas: 7 en 0,094 segundos

	ID_PASAJERO	NOMBRE_PASAJERO	CORREO_PASAJERO	TELEFONO_PASAJERO	ID_SOLICITUD	TIPO_SERVICIO	NIVEL_SERVICIO	ESTADO_SOLICITUD	ID_VIAJE	FECHA_ASIGNACION	HORA_INICIO	HORA_FIN	DISTANCIA_KM	COSTO_TOTAL
1	24	Pasajero_24	pasajero24@alpescab.com	3005739055	822	MERCANCIAS	ESTANDAR	CREADA	822	21/07/24	21/07/24	21/07/24	9,36	12327
2	24	Pasajero_24	pasajero24@alpescab.com	3005739055	42	PASAJEROS	LARGE	CREADA	42	18/03/24	18/03/24	18/03/24	7,4	12335
3	24	Pasajero_24	pasajero24@alpescab.com	3005739055	747	PASAJEROS	ESTANDAR	CREADA	747	20/02/24	20/02/24	20/02/24	5,63	9326
4	24	Pasajero_24	pasajero24@alpescab.com	3005739055	367	COMIDA	CONFORT	CREADA	367	04/02/24	04/02/24	04/02/24	6,79	11005
5	24	Pasajero_24	pasajero24@alpescab.com	3005739055	766	MERCANCIAS	CONFORT	CREADA	766	31/12/23	31/12/23	31/12/23	5,65	8056
6	24	Pasajero_24	pasajero24@alpescab.com	3005739055	905	PASAJEROS	ESTANDAR	CREADA	905	05/11/23	05/11/23	05/11/23	14,02	18249
7	24	Pasajero_24	pasajero24@alpescab.com	3005739055	379	COMIDA	CONFORT	CREADA	379	02/11/23	02/11/23	02/11/23	0,5	3024

RFC2 Top 20 conductores

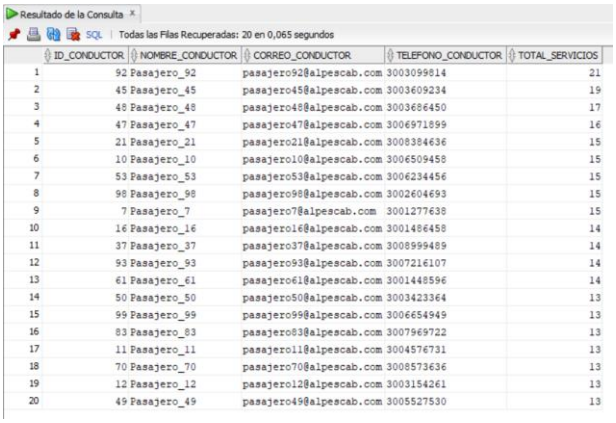
ID	Objetivo	Datos de entrada	Pasos	Resultado esperado
RFC2 A	Ranking por número de viajes	Viajes varios por 2001 y 2002	Agrupar por conductor y contar viajes, ordenar desc, limitar 20	2001 aparece por encima de 2002 si tiene más viajes
RFC2 B	Empates y bordes	20 o más conductores	Ejecutar ranking	Lista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

La consulta muestra los 20 conductores que más servicios han prestado en la aplicación. Se seleccionan los datos básicos del conductor desde la tabla **USUARIO_SERVICIO** (id, nombre, correo, teléfono) y se cuentan los viajes realizados desde la tabla **VIAJE** mediante un `COUNT` sobre `id_viaje`. El `JOIN` se hace con la relación `id_usuario_servicio = id_usuario_conductor`, ya que cada viaje está asociado a un conductor. Después, se agrupa con `GROUP BY` para calcular el total de servicios por conductor, se ordena en orden descendente según la cantidad de servicios y finalmente se limita el resultado a los 20 primeros.

Las tablas involucradas son: **USUARIO_SERVICIO**, que contiene la información de los usuarios de la aplicación (en este caso, los conductores), y **VIAJE**, que registra cada viaje realizado con atributos como `id_viaje`, fechas, costo, distancia, id del vehículo, id de la solicitud y el `id_usuario_conductor` que lo presta.

Se utilizó un **INNER JOIN** porque un viaje siempre debe estar asociado a un conductor válido; no tendría sentido mostrar conductores que no tengan viajes asociados en este análisis. De esta manera se garantiza que en el resultado aparezcan solo aquellos que han prestado al menos un servicio.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148



Resultado de la Consulta

Todas las Filas Recuperadas: 20 en 0,065 segundos

	ID_CONDUCTOR	NOMBRE_CONDUCTOR	CORREO_CONDUCTOR	TELEFONO_CONDUCTOR	TOTAL_SERVICIOS
1	92	Pasajero_92	pasajero92@alpescab.com	3003099814	21
2	45	Pasajero_45	pasajero45@alpescab.com	3003609234	19
3	48	Pasajero_48	pasajero48@alpescab.com	3003686450	17
4	47	Pasajero_47	pasajero47@alpescab.com	3006971899	16
5	21	Pasajero_21	pasajero21@alpescab.com	3008384636	15
6	10	Pasajero_10	pasajero10@alpescab.com	3006509458	15
7	53	Pasajero_53	pasajero53@alpescab.com	3006234456	15
8	98	Pasajero_98	pasajero98@alpescab.com	3002604693	15
9	7	Pasajero_7	pasajero7@alpescab.com	3001277638	15
10	16	Pasajero_16	pasajero16@alpescab.com	3001486458	14
11	37	Pasajero_37	pasajero37@alpescab.com	3008999489	14
12	93	Pasajero_93	pasajero93@alpescab.com	3007216107	14
13	61	Pasajero_61	pasajero61@alpescab.com	3001448596	14
14	50	Pasajero_50	pasajero50@alpescab.com	3003423364	13
15	99	Pasajero_99	pasajero99@alpescab.com	3006654949	13
16	83	Pasajero_83	pasajero83@alpescab.com	3007969722	13
17	11	Pasajero_11	pasajero11@alpescab.com	3004576731	13
18	70	Pasajero_70	pasajero70@alpescab.com	3008573636	13
19	12	Pasajero_12	pasajero12@alpescab.com	3003154261	13
20	49	Pasajero_49	pasajero49@alpescab.com	3005527530	13

RFC3 Dinero ganado por vehículo y servicio

ID	Objetivo	Datos de entrada	Pasos	Resultado esperado
RFC3 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
RFC3 B	Rango de fechas	Fechas cruzadas	Filtrar por rango	Sumas se ajustan al rango dado

La consulta calcula el total de dinero obtenido por cada conductor en cada uno de sus vehículos, discriminado por tipo de servicio. Se toman los datos del conductor desde **USUARIO_SERVICIO** (id y nombre), los datos del vehículo desde **VEHICULO** (id, placa, marca, modelo), y el tipo de servicio desde **SOLICITUD_SERVICIO**. El monto se obtiene sumando `costo_total` de cada viaje en la tabla **VIAJE**, aplicando la deducción de la comisión de la plataforma (aquí se asumió 20%, quedando 80% para el conductor). Se agrupa por conductor, vehículo y tipo de servicio, de modo que el resultado muestra cuánto ha ganado un conductor por cada servicio específico con cada vehículo.

Las tablas involucradas son: **USUARIO_SERVICIO**, con la información de los conductores; **VIAJE**, que registra los viajes realizados con su costo y referencia al conductor, vehículo y solicitud; **VEHICULO**, que identifica el automóvil utilizado; y **SOLICITUD_SERVICIO**, que define el tipo de servicio solicitado.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

Se usaron **INNER JOINs** porque todos los viajes considerados deben tener un conductor, un vehículo y una solicitud asociada; no tendría sentido sumar viajes incompletos en este reporte financiero. Así se asegura que los datos mostrados correspondan únicamente a servicios válidos y completos.

Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 287 en 0,387 segundos

	ID_CONDUCTOR	NOMBRE_CONDUCTOR	ID_VEHICULO	PLACA	MARCA	MODELO	TIPO_SERVICIO	TOTAL_GANADO
1	1	Pasajero_1	1	ZX280	Ford	2018	COMIDA	27808,8
2	1	Pasajero_1	1	ZX280	Ford	2018	MERCANCIAS	31624,8
3	1	Pasajero_1	1	ZX280	Ford	2018	PASAJEROS	13377,6
4	2	Pasajero_2	2	ZD787	Mazda	2014	COMIDA	13166,4
5	2	Pasajero_2	2	ZD787	Mazda	2014	MERCANCIAS	38564
6	2	Pasajero_2	2	ZD787	Mazda	2014	PASAJEROS	20026,4
7	3	Pasajero_3	3	SZ369	Hyundai	2018	COMIDA	3385,6
8	3	Pasajero_3	3	SZ369	Hyundai	2018	MERCANCIAS	34958,4
9	3	Pasajero_3	3	SZ369	Hyundai	2018	PASAJEROS	11953,6
10	4	Pasajero_4	4	JW826	Hyundai	2014	COMIDA	44219,2
11	4	Pasajero_4	4	JW826	Hyundai	2014	MERCANCIAS	30116
12	4	Pasajero_4	4	JW826	Hyundai	2014	PASAJEROS	27533,6
13	5	Pasajero_5	5	DP585	Honda	2016	COMIDA	54468
14	5	Pasajero_5	5	DP585	Honda	2016	MERCANCIAS	68887,2
15	5	Pasajero_5	5	DP585	Honda	2016	PASAJEROS	24504
16	6	Pasajero_6	6	QG324	Toyota	2015	COMIDA	38828
17	6	Pasajero_6	6	QG324	Toyota	2015	MERCANCIAS	3148,8
18	6	Pasajero_6	6	QG324	Toyota	2015	PASAJEROS	24350,4
19	7	Pasajero_7	7	EF757	Ford	2013	COMIDA	51141,6
20	7	Pasajero_7	7	EF757	Ford	2013	MERCANCIAS	71566,4
21	7	Pasajero_7	7	EF757	Ford	2013	PASAJEROS	26336
22	8	Pasajero_8	8	ID948	Mazda	2010	COMIDA	29276,8
23	8	Pasajero_8	8	ID948	Mazda	2010	MERCANCIAS	31500
24	8	Pasajero_8	8	ID948	Mazda	2010	PASAJEROS	49552,8
25	9	Pasajero_9	9	SH446	Toyota	2017	COMIDA	47900
26	9	Pasajero_9	9	SH446	Toyota	2017	MERCANCIAS	37616

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

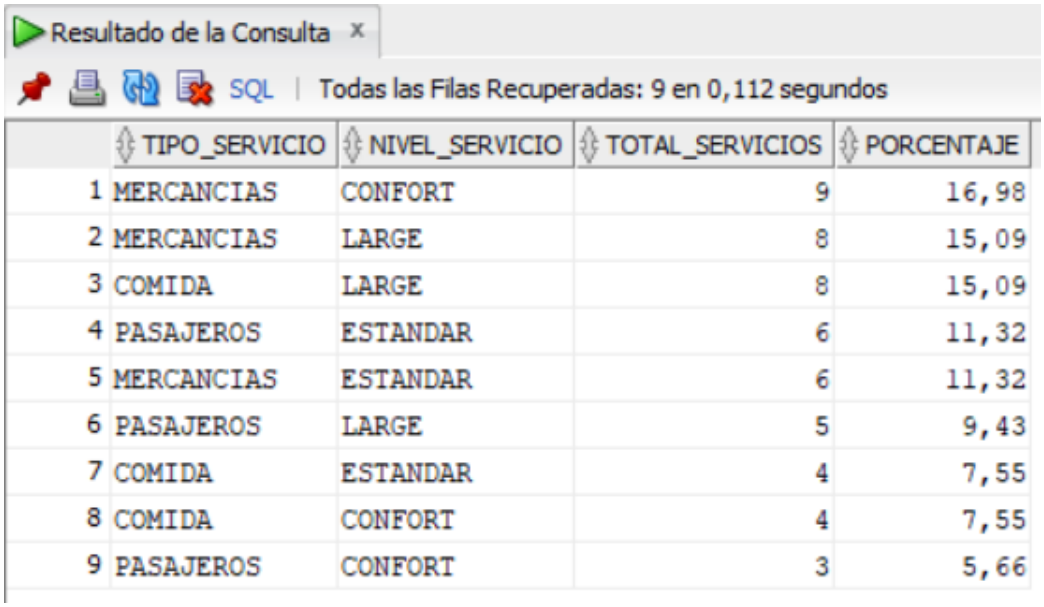
Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

La consulta permite visualizar la utilización de los servicios de AlpesCab en una ciudad durante un rango de fechas dado. Se cuentan todos los viajes (`COUNT(v.id_viaje)`) agrupados por tipo de servicio y nivel tomados de la tabla **SOLICITUD_SERVICIO**. A la vez, se calcula el porcentaje que representa cada grupo respecto al total de servicios realizados, usando una ventana `SUM(COUNT(...)) OVER()`. El resultado se ordena de mayor a menor cantidad de servicios para mostrar primero los más usados.

Las tablas utilizadas son: **SOLICITUD_SERVICIO**, que indica el tipo (`PASAJEROS`, `COMIDA`, `MERCANCIAS`) y el nivel (`ESTANDAR`, `CONFORT`, `LARGE`) del servicio; **VIAJE**, que registra cada viaje realizado, incluyendo fecha de asignación para filtrar por el rango indicado; **VEHICULO**, que enlaza con la ciudad de expedición; y **CIUDAD**, que permite seleccionar el análisis para una ciudad específica.

Se aplicaron **INNER JOINs** porque solo se consideran viajes válidos que tengan solicitud, vehículo y ciudad asociada, ya que la finalidad es obtener estadísticas completas y confiables de servicios efectivamente prestados.

La siguiente imagen es el resultado de la consulta tomando como ciudad “Bogotá” y como fechas “2024-01-01” a “2024-12-31”.


Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 9 en 0,112 segundos

	TIPO_SERVICIO	NIVEL_SERVICIO	TOTAL_SERVICIOS	PORCENTAJE
1	MERCANCIAS	CONFORT	9	16,98
2	MERCANCIAS	LARGE	8	15,09
3	COMIDA	LARGE	8	15,09
4	PASAJEROS	ESTANDAR	6	11,32
5	MERCANCIAS	ESTANDAR	6	11,32
6	PASAJEROS	LARGE	5	9,43
7	COMIDA	ESTANDAR	4	7,55
8	COMIDA	CONFORT	4	7,55
9	PASAJEROS	CONFORT	3	5,66

4.1.1. Matriz de cobertura

Req	Caso OK	Duplicado/ UNIQUE	FK inexistente	Check/Regla negocio	Observación

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

RF1 Ciudad	✓	nombre ya existe → 409	—	nombre vacío → 400	Autonumeración habilitada
RF2 UsuarioServicio	✓	id/correo/cedula repetidos → 409	—	formatos básicos	—
RF3 Conductor	✓	id/correo/cedula repetidos → 409	—	0 ≤ comisión ≤ 1	—
RF4 Vehículo	✓	placa o id duplicado → 409	ciudad/conductor inexistente → 404	—	Nivel **no** aplica en vehículo
RF5 Disponibilidad	✓	id duplicado → 409	vehículo/conductor inexistente → 404	solape horario → 409; día/tipo inválido → 400	`hora_fin > hora_inicio`
RF6 Disponibilidad (otro tipo)	✓	—	—	mismas que RF5	—
RF7 Punto geográfico	✓	id duplicado → 409	ciudad inexistente → 404	lat/long fuera de rango → 400 (si se valida)	—
RF8a Solicitud	✓	id duplicado → 409	usuario/punto inexistente → 404	fecha inválida → 400	`nivel` en Solicitud
RF8b Viaje	✓	id duplicado → 409	vehículo/conductor/punto/solicitud inexistente → 404	solicitud con viaje previo → 409	admite 0 en métricas
RF9 Finalizar viaje	✓	—	viaje inexistente → 404	falta `horaInicio` si estaba NULL → 400; `horaFin ≤ horaInicio` → 400	—
RF10 Reseña cliente→conductor	✓	id duplicado → 409	viaje inexistente → 404	calificación fuera de [0..5] → 400	—
RF11 Reseña conductor →cliente	✓	id duplicado → 409	viaje inexistente → 404	calificación fuera de [0..5] → 400	—
RFC1 Histórico usuario-servicio	✓	—	usuario inexistente → 404 (si se valida)	rango fechas invertido → 400 (si se valida)	Ordenable por fecha

RFC2 Top conductores	✓	—	—	rango vacío devuelve lista vacía	Top 20
RFC3 Ganancias por conductor	✓	—	conductor inexistente → 404 (si se valida)	—	Agrupa por vehículo/tipo
RFC4 Utilización por ciudad	✓	—	ciudad inexistente → 404 (si se valida)	—	% respecto al total del rango

- Roles USR cliente = UsuarioServicio, COND = UsuarioConductor, ADM = Administrador

4.2. Pruebas de corrección y calidad de datos

4.2.1 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

4.2.2 Integridad referencial con FKs

ID	Objetivo	Tabla hija	FK	Pre condiciones	Caso	Pasos	Resultado esperado

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

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	

4.2.3 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

4.2.4 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

4.3. Criterios de aceptación generales

- Todas las rutas RF1–RF11 / RFC1–RFC4 devuelven 200 en el escenario feliz con datos válidos.
- Las violaciones de integridad/negocio retornan 400/404/409 con mensajes claros.

Mariana Cediel – 202321548
Alejandro Cruz – 201912149
Nicolás Hernández - 202322148

- Los reportes responden en tiempos aceptables (muestras ≈ miles de filas) y devuelven resultados consistentes con los datos cargados.
- La plantilla Excel del modelo relacional corresponde al esquema implementado y a las validaciones realizadas.