Bases de Datos

Segundo Cuatrimestre de 2011

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Trabajo Práctico

Primera parte

Integrante	LU	Correo electrónico
Hernan Bandura	416/03	hbandura@gmail.com
Fabian González	076/01	fsgonz@yahoo.com.ar
Santiago Perez	915/03	binarius@gmail.com
Francisco Roslan	356/03	panchis@gmail.com

Índice

1. Introducción

El presente documento describe y analiza la solución propuesta por el group de trabajo al problema descripto en el enunciado "Bases de Datos - Segundo cuatrimestre 2011 / Trabajo práctico - Primera Parte" presentado por la cátedra.

1.1. Organización del documento

El desarrollo de la solución se dividió en las etapas de análisis, modelado, implementación y testing, que son detalladas en las siguientes secciones del documento.

2. Análisis

En esta sección se analiza el documento de enunciado presentado por la cátedra. De este documento se extraen las características fundamentales y necesarias que se espera de la solución. La identificación y organización de estos requerimientos, es de gran utilidad para verificar que la solución planteada satisface las caractéristicas del problema y contiene los entregables solicitados .

2.1. Entregables

Los entregables solicitados se identifican en la sección "Consignas de la primera parte" del enunciado

1. Modelado
   - I Modelo de Entidad-Relación: item a)
   - II Modelo Relacional: item a)
   - III Detalles del modelo y verificación: items b) y d)
2. Implementación
   - I Diseño físico: item c)
   - II Procedimientos y triggers: item e)
3. Testing
   - I Datos de prueba: segundo párrafo
   - II Casos de prueba: segundo párrafo

2.2. Requerimientos

Los requerimientos son las características fundamentales y esperadas que la solución debe satisfacer y se identifican en la sección "Descripción del problema".

2.2.1. Resumen

El enunciado describe como se relacionan los recursos humanos y materiales de una empresa de transporte y plantea la necesidad de un sistema informático que ayude a administrar dicha empresa. A continuación se detallan los requerimientos sobre este sistema que surgen de la descripción del problema.

2.2.2. Lista detallada

1. Registros principales: el sistema debe ser capaz de registrar los datos que a la empresa le interesa mantener de sus choferes, sus vehículos y los recorridos que ofrece. Estos registros son la base para administrar y controlar los viajes y las polítcas de la empresa.

I Registro de choferes
- Datos personales: apellido, nombre, documento, fecha de nacimiento, domicilio y teléfono.
- Datos de licencia de conducir: número, tipo, fecha de otorgamiento, fecha de renovación, fecha vencimiento y observaciones.

II Registro de vehículos
- Datos del vehículo: número de patente, marca, modelo.
- Datos relacionados al servicio: antiguedad en el servicio, capacidad de transporte y estado.
- Datos relacionados al estado: El estado de un vehículo puede ser "en uso" o "en reparación". De estar el vehículo en reparación se registra la fecha de ingreso al taller.

III Registro de recorridos
- Datos de origen y destino: ciudad, calle y número.
- Los recorridos pueden tener una o más rutas posibles que unen el origen y el destino.
- Datos de las rutas: descripción, largo, tiempo estimado del recorrido, peajes, condiciones del camino y condiciones climáticas (según el período del año).

2. Control de viajes: el sistema debe ser capaz de ayudar a administrar los viajes que la empresa realiza y sus controles.

I Viajes planificados
- Datos a registrar: recorrido, fecha y hora de partida, fecha y hora de llegada estimada, vehículo y choferes asignados
- Los viajes los planifica la empresa con los recursos que cuenta, como ser vehículos en uso y choferes habilitados.
- Cada viaje tiene un máximo de tres choferes asignados.

II Viajes realizados
- Datos a registrar: fecha y hora de llegada real, ruta elegida y si hubo problemas o demoras en el camino.
- Un mismo recorrido podra ser realizado peridicamente por el mismo vehculo y los mismos choferes en distintas fechas.

III Registro de controles
- Datos: tipo de test (visión, alcoholemia, ect.), chofer, viaje relacionado, fecha de realizacion y resultado.
- Los controles se hacen a choferes asingados a viajes planificados, antes de la realización del mismo.

3. Procedimientos y políticas El sistema debe satistfacer las funcionalidades identificadas en la sección "Comentario de la cátedra"

I Recorridos para los cuales se usaron todas las rutas asociadas en viajes del año pasado

II Recorridos con mas de una ruta asociada

III Estado y viajes realizados por vehículo

IV Lista de choferes que han utilizado, en el último semestre, todos los vehículos de menos de dos años de antiguedad.

V Implemtación de alguna restricción adicional que surja del diseño.

3. Modelado

3.1. Choferes

Las siguientes entidades abarcan los requerimientos listados como 1.I

Chofer (DNI, apellido, nombre, fechaNacimiento, domicilio, telefono)
PK = {DNI}
CK = {DNI}

Licencia (NUMEROLICENCIA, fechaOtorgamiento, tipo, DNI, fechaRenovacion, fechaVencimiento, observaciones)
PK = {(NUMEROLICENCIA, fechaOtorgamiento, tipo)}
CK = {(NUMEROLICENCIA, fechaOtorgamiento, tipo), (NUMEROLICENCIA, fechaVencimiento, tipo), (DNI, fechaOtorgamiento, tipo), (DNI, fechaVencimiento, tipo) }
FK = {DNI}

Restricciones

- *Licencia.DNI* "debe estar en" *Chofer.DNI*
- *Licencia.fechaOtorgamiento* debe ser anterior a *Licencia.fechaRenovacion*
- *Licencia.fechaRenovacion* debe ser anterior a *Licencia.fechaVencimiento*

3.2. Vehículos

Las siguientes entidades y restriccciones abarcan los requerimientos listados como 1.II

Vehiculo (PATENTE, IDMODELO, IDMARCA, año, fechaInicioServicio, capacidadDeTransporte, situacion)
PK = CK = {PATENTE}
FK = {IDMODELO, IDMARCA}

VehiculoEnReparacion (PATENTE, fechaIngresoTaller)
PK = CK = FK = {PATENTE}

Modelo (IDMODELO, IDMARCA, nombre)
PK = CK = {(IDMODELO, IDMARCA)}
FK = {IDMARCA}

Marca (IDMARCA, nombre)
PK = CK = {IDMARCA}
FK = {}

Restricciones

- *VehiculoEnUso.PATENTE* "debe estar en" *Vehiculo.PATENTE*
- *VehiculoEnReparacion.PATENTE* "debe estar en" *Vehiculo.PATENTE*
- *Vehiculo.ano* debe ser anterior o igual al año de *Vehiculo.fechaInicioServicio*
- *Vehiculo.PATENTE "debe estar en" VehiculoEnUso.PATENTE o (exclusivo) VehiculoEnReparacion.PATENTE*

  (Vehiculo.IDMODELO, Vehiculo.IDMARCA) "debe ser PK de" *Modelo.IDMODELO*
- *Modelo.IDMARCA* "debe estar en" *Marca.IDMARCA*
- El atributo *Vehiculo.situacion* permite particionar en forma disjunta el conjunto de vehículos.

3.3. Recorridos y rutas

Las siguientes entidades abarcan los requerimientos listados como 1.III

Ciudad (IDCIUDAD, nombre)
PK = CK = {IDCIUDAD}

Recorrido (IDRECORRIDO, IDCIUDADORIGEN, direccionOrigen, IDCIUDADDESTINO, direccionDestino)
PK = {IDRECORRIDO}
CK = {IDRECORRIDO, (IDCIUDADORIGEN, direccionOrigen, IDCIUDADDESTINO, direccionDestino)}
FK = {IDCIUDADORIGEN, IDCIUDADDESTINO}

Ruta (IDRUTA, IDRECORRIDO, longitud, tiempoEstimado, cantidadPeajes, descripcion, observaciones)
PK = CK = {(IDRUTA, IDRECORRIDO)}
FK = { IDRECORRIDO }

CondicionClimatica (IDCONDICIONCLIMATICA, descripcion)
PK = CK = {IDCONDICIONCLIMATICA}

PeriodoAño (IDPERIODOAÑO, descripcion)
PK = CK = {IDPERIODOAÑO}

SituacionClimaticaRuta (IDRUTA, IDPERIODOAÑO, IDCONDICIONCLIMATICA)
PK = CK = { (IDRUTA, IDPERIODOAÑO, IDCONDICIONCLIMATICA)}
FK = { IDRUTA, IDPERIODOAÑO, IDCONDICIONCLIMATICA }

Restricciones

- *Recorrido.IDCIUDADORIGEN* "debe estar en" *Ciudad.IDCIUDAD*
- *Recorrido.IDCIUDADDESTINO* "debe estar en" *Ciudad.IDCIUDAD*
- *Ruta.IDRECORRIDO* "debe estar en" *Recorrido.IDRECORRIDO*

3.4. Viajes y controles

Las siguientes entidades y restriccciones abarcan los requerimientos listados como 2.I, 2.II y 2.III

Viaje (IDVIAJE, IDVEHICULO, IDRECORRIDO, fechaHoraPartida, fechaHoraLlegadaEstimada)
PK = {IDVIAJE}
CK = {IDVIAJE, (IDVEHICULO, fechaHoraPartida)}
FK = {IDVEHICULO, IDRECORRIDO}

ViajeRealizado (IDVIAJE, IDRUTA, fechaHoraLlegada, contingencias)
PK = CK = {IDVIAJE}
FK = {IDRUTA}

AsignacionChoferViaje (IDVIAJE, DNI)
PK = CK={(DNI, IDVIAJE)}
FK = {DNI, IDVIAJE}

TipoTest (IDTIPOTEST, descripcion)
PK = CK = {IDTIPOTEST}

Test (IDVIAJE, DNI, IDTIPOTEST, fechaRealizacion, resultado)
PK = CK ={(DNI, IDVIAJE, IDTIPOTEST)}
FK = {DNI, IDVIAJE, IDTIPOTEST}

Restricciones

- *Viaje.IDRECORRIDO* "debe estar en" *Recorrido.IDRECORRIDO*
- Viaje.IDVEHICULO debe estar en uso.
- *ViajeRealizado.IDRUTA* "debe estar en" *Ruta.IDRUTA*
- *ViajeRealizado.IDRUTA y Viaje.IDRECORRIDO para el mismo viaje* "debe estar en" *(Ruta.IDRUTA, Ruta.IDRECORRIDO)*
- *ViajeRealizado.IDVIAJE* "debe estar en" *Viaje.IDVIAJE*
- *AsignacionChofer.DNI* "debe estar en" *Chofer.DNI*
- *Test.DNI* "debe estar en" *Chofer.DNI*
- *Test.IDTIPOTEST* "debe estar en" *TipoTest.IDTIPOTEST*
- *Test.IDVIAJE* "debe estar en" *Viaje.IDVIAJE*
- la ruta de un *ViajeRealizado* debe pertener al recorrido del *Viaje* con el mismo IDVIAJE.
- *AsignacionChoferViaje.IDVIAJE* "no debe repetirse mas de tres veces"
- *Test.fechaRealizacion* "debe ser anterior a" *Viaje.fechaPartida* "cuando" *Test.IDVIAJE = Viaje.IDVIAJE*

4. Implementación

4.1. Motor de bases de datos

El motor de base de datos utilizado para implementar la solución es MySQL. Fue elegido por ser el más conocido por el equipo de trabajo y por ser un software de bases de datos libre con una importante comunidad de usuarios.
Lo que sigue en esta sección es el código fuente para la creación del modelo. Estos códigos fuentes tambien se pueden encontrar en el directorio "impl/" de la entrega.

4.2. Choferes

```
CREATE TABLE  `choferes` (
  `dni` varchar(10) NOT NULL,
  `telefono` varchar(20) NOT NULL,
  `fecha_nacimiento` date NOT NULL,
  `domicilio` varchar(30) NOT NULL,
  `nombre` varchar(100) NOT NULL,
  `apellido` varchar(100) NOT NULL,
  PRIMARY KEY (`dni`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `licencias` (
  `nro_licencia` int(11) NOT NULL,
  `fecha_otorgamiento` date NOT NULL,
  `tipo` varchar(100) NOT NULL,
  `observaciones` varchar(100) NOT NULL,
  `fecha_renovacion` datetime NOT NULL,
  `dni_chofer` varchar(10) NOT NULL,
  `fecha_vencimiento` datetime NOT NULL,
  PRIMARY KEY (`nro_licencia`,`fecha_otorgamiento`,`tipo`),
  KEY `dnis_constraint` (`dni_chofer`),
  CONSTRAINT `dnis_constraint` FOREIGN KEY (`dni_chofer`) REFERENCES `choferes` (`dni`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

4.3. Vehículos

```
CREATE TABLE  `marcas` (
  `id_marca` int(11) NOT NULL,
  `nombre` varchar(100) NOT NULL,
  PRIMARY KEY (`id_marca`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE  `bases`.`modelos` (
  `id_modelo` int(11) NOT NULL,
  `nombre` varchar(30) NOT NULL,
  `id_marca` int(11) NOT NULL,
  PRIMARY KEY (`id_modelo`,`id_marca`) USING BTREE,
  KEY `marca_fk_constraint` (`id_marca`),
  CONSTRAINT `marca_fk_constraint` FOREIGN KEY (`id_marca`) REFERENCES `marcas` (`id_marca`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `vehiculos` (
  `patente` varchar(10) NOT NULL,
  `ano` int(11) NOT NULL,
```

```
  `situacion` tinyint(1) NOT NULL,
  `id_modelo` int(11) NOT NULL,
  `id_marca` int(11) NOT NULL,
  `fecha_inicio_servicio` datetime NOT NULL,
  PRIMARY KEY (`patente`),
  KEY `veh_modelo_constraint` (`id_modelo`,`id_marca`),
  CONSTRAINT `veh_modelo_constraint` FOREIGN KEY (`id_modelo`, `id_marca`)
  REFERENCES `modelos` (`id_modelo`, `id_marca`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=DYNAMIC;

CREATE TABLE  `vehiculos_en_reparacion` (
  `patente` varchar(10) NOT NULL,
  `fecha_ingreso_taller` datetime NOT NULL,
  PRIMARY KEY (`patente`),
  CONSTRAINT `patenet_fk_constraint` FOREIGN KEY (`patente`) REFERENCES `vehiculos` (`patente`)
  ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

4.4. Recorridos y rutas

```
CREATE TABLE  `ciudades` (
  `id_ciudad` int(11) NOT NULL,
  `nombre_ciudad` varchar(30) NOT NULL,
  PRIMARY KEY (`id_ciudad`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `recorridos` (
  `id_recorrido` int(11) NOT NULL,
  `id_ciudad_origen` int(11) NOT NULL,
  `id_ciudad_destino` int(11) NOT NULL,
  `direccion_origen` varchar(100) NOT NULL,
  `direccion_destino` varchar(100) NOT NULL,
  PRIMARY KEY (`id_recorrido`),
  KEY `ciudad_origen_constraint` (`id_ciudad_origen`),
  KEY `ciudad_destino_constraint` (`id_ciudad_destino`),
  CONSTRAINT `ciudad_destino_constraint` FOREIGN KEY (`id_ciudad_destino`) REFERENCES `ciudades` (`i
  CONSTRAINT `ciudad_origen_constraint` FOREIGN KEY (`id_ciudad_origen`) REFERENCES `ciudades` (`id_
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE  `rutas` (
  `id_ruta` int(11) NOT NULL,
  `id_recorrido` int(11) NOT NULL,
  `longitud` int(11) NOT NULL,
  `tiempo_estimado` int(11) NOT NULL,
  `cantidad_peajes` int(11) NOT NULL,
  `descripcion` varchar(100) NOT NULL,
  PRIMARY KEY (`id_ruta`,`id_recorrido`) USING BTREE,
  KEY `recorrido_constraint` (`id_recorrido`),
  CONSTRAINT `recorrido_constraint` FOREIGN KEY (`id_recorrido`)
  REFERENCES `recorridos` (`id_recorrido`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE  `condiciones_climaticas` (
  `id_condicion_climatica` int(11) NOT NULL,
  `descripcion` varchar(100) NOT NULL,
  PRIMARY KEY (`id_condicion_climatica`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE  `periodos_ano` (
  `id_periodo_ano` int(11) NOT NULL,
  `descripcion` varchar(100) NOT NULL,
  PRIMARY KEY (`id_periodo_ano`) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `situaciones_climaticas` (
  `id_ruta` int(11) NOT NULL,
  `id_condicion_climatica` int(11) NOT NULL,
  `id_periodo_ano` int(11) NOT NULL,
  PRIMARY KEY (`id_ruta`,`id_condicion_climatica`,`id_periodo_ano`) USING BTREE,
  KEY `condicion_climatica_constraint` (`id_condicion_climatica`),
  CONSTRAINT `condicion_climatica_constraint`
   FOREIGN KEY (`id_condicion_climatica`) REFERENCES `condiciones_climaticas` (`id_condicion_climati
  CONSTRAINT `id_ruta_constraint` FOREIGN KEY (`id_ruta`) REFERENCES `rutas` (`id_ruta`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

4.5. Viajes y Controles

```
CREATE TABLE  `viajes` (
  `id_viaje` int(11) NOT NULL,
  `id_vehiculo` varchar(10) NOT NULL,
  `contingencias` varchar(300) NOT NULL,
  `fecha_hora_partida` datetime NOT NULL,
  `id_recorrido` int(11) NOT NULL,
  `fecha_hora_llegada_estimada` datetime NOT NULL,
  PRIMARY KEY (`id_viaje`),
  KEY `new_fk_constraint` (`id_vehiculo`),
  KEY `recorridos_constraint` (`id_recorrido`),
  CONSTRAINT `new_fk_constraint` FOREIGN KEY (`id_vehiculo`) REFERENCES `vehiculos` (`patente`),
  CONSTRAINT `recorridos_constraint` FOREIGN KEY (`id_recorrido`)
  REFERENCES `recorridos` (`id_recorrido`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `viajes_realizados` (
  `id_viaje` int(11) NOT NULL,
  `fecha_hora_llegada` datetime NOT NULL,
  `id_ruta` int(11) NOT NULL,
  PRIMARY KEY (`id_viaje`),
  KEY `rutas_constraint` (`id_ruta`),
  CONSTRAINT `rutas_constraint` FOREIGN KEY (`id_ruta`) REFERENCES `rutas` (`id_ruta`),
  CONSTRAINT `viajes_realizados_constraint` FOREIGN KEY (`id_viaje`)
  REFERENCES `viajes` (`id_viaje`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `test` (
  `id_tipo_test` int(11) NOT NULL,
  `dni` varchar(10) NOT NULL,
  `id_viaje` int(11) NOT NULL,
  `resultado` varchar(100) NOT NULL,
  `fecha_realizacion` datetime NOT NULL,
  PRIMARY KEY (`id_tipo_test`,`dni`,`id_viaje`),
  KEY `viaje_test_constraint` (`id_viaje`),
  KEY `dni_test_constraint` (`dni`),
  CONSTRAINT `dni_test_constraint` FOREIGN KEY (`dni`)
  REFERENCES `choferes` (`dni`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `tipo_test_constraint` FOREIGN KEY (`id_tipo_test`)
```

```
  REFERENCES `tipo_test` (`id_tipo_test`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `viaje_test_constraint` FOREIGN KEY (`id_viaje`)
  REFERENCES `viajes` (`id_viaje`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `tipo_test` (
  `id_tipo_test` int(11) NOT NULL,
  `descripcion` varchar(100) NOT NULL,
  PRIMARY KEY (`id_tipo_test`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE  `asignaciones_chofer` (
  `dni` varchar(10) NOT NULL,
  `id_viaje` int(11) NOT NULL,
  PRIMARY KEY (`dni`,`id_viaje`),
  KEY `viajes_asignacion_constraint` (`id_viaje`),
  CONSTRAINT `dni_asignacion_constraint` FOREIGN KEY (`dni`)
  REFERENCES `choferes` (`dni`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `viajes_asignacion_constraint` FOREIGN KEY (`id_viaje`)
  REFERENCES `viajes` (`id_viaje`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

4.6. Stored Procedures

RecorridosTodasRutasUsadas

El siguiente *stored procedure* obtiene todos los recorridos para los cuales se usaron todas las rutas posibles registradas para ese recorrido, para viajes realizados el ao pasado y recorridos asociados a ms de una ruta y corresponde al requerimiento listado como 3.I. Conceptualmente, se trata de una división. Se obtienen todos los recorridos, tales que no existen rutas de ese recorrido para las que no ha habido viajes del último año que la usaran. El último ao se pasa como parámetro y se usan funciones auxiliares:

```
CREATE DEFINER=`root`@`localhost` PROCEDURE
`RecorridosTodasRutasUsadas`(IN ano integer)
BEGIN
       SELECT r.id_recorrido
               FROM recorridos r
               WHERE NOT EXISTS
 (SELECT *
  FROM rutas rut
  WHERE rut.id_recorrido = r.id_recorrido AND NOT EXISTS (
SELECT *
 FROM viajes_realizados v
 NATURAL JOIN viajes v1
 WHERE YEAR(v.fecha_hora_llegada) = ano
       AND v.id_ruta = rut.id_ruta
       AND v1.id_recorrido = r.id_recorrido
)
 )
      AND ((select cantidadRutasPorRecorrido(r.id_recorrido)) > 1);
END

CREATE DEFINER=`root`@`localhost` FUNCTION
`cantidadRutasPorRecorrido`(id_recorrido integer) RETURNS int(11)
begin
    RETURN (SELECT count(ruta.id_ruta) FROM rutas ruta WHERE ruta.id_recorrido = id_recorrido);
end
```

El siguiente *stored procedure* obtiene el promedio de viajes realizados por vehículo por año y el estado en que éste se encuentra, corresponde al requerimiento listado como 3.III.

```
CREATE DEFINER=‘root‘@‘localhost‘ FUNCTION  ‘bases‘.‘promedioViajesRealizadosPorAno‘
(patente varchar(10)) RETURNS int(11)
begin
    DECLARE ano_inicio_servicio int;
    DECLARE viajes_realizados int;
    SELECT count(*) into viajes_realizados FROM viajes_realizados
     NATURAL JOIN viajes v WHERE  v.id_vehiculo = patente;
    SELECT YEAR(fecha_inicio_servicio)  into ano_inicio_servicio
     FROM vehiculos v where v.patente = patente;
    RETURN (viajes_realizados / (YEAR(NOW()) - ano_inicio_servicio + 1)) ;
end

CREATE DEFINER=‘root‘@‘localhost‘ PROCEDURE  ‘bases‘.‘promedioViajesYEstado‘()
BEGIN
SELECT (SELECT promedioViajesRealizadosPorAno(patente)), situacion from vehiculos;
END
```

El siguiente *stored procedure* obtiene los choferes que han utilizado todos los vehículos de menos de dos años de antiguedad, en viajes del último semestre. Corresponde al requerimiento listado como 3.IV.

```
CREATE DEFINER=‘root‘@‘localhost‘ PROCEDURE
‘ChoferesTodosLosVehiculosUltimoSemestre‘()
BEGIN
       SELECT *
               FROM choferes c
               WHERE NOT EXISTS
 (SELECT *
  FROM vehiculos v
  WHERE (SELECT agregardosanos(v.patente)) > NOW() AND NOT EXISTS (
SELECT *
 FROM viajes_realizados v1
 NATURAL JOIN viajes v2
 WHERE (SELECT agregarseismeses(v1.fecha_hora_llegada)) > NOW()
       AND v2.id_vehiculo = v.patente
 AND EXISTS (
      SELECT * FROM asignaciones_chofer WHERE
 dni = c.dni AND id_viaje = v2.id_viaje
)
)
 );
END

CREATE DEFINER=‘root‘@‘localhost‘ FUNCTION
‘agregardosanos‘(id_vehiculo varchar(10)) RETURNS datetime
BEGIN
return DATE_ADD((SELECT fecha_inicio_servicio FROM vehiculos where patente = id_vehiculo),
INTERVAL 2 YEAR);
END


CREATE DEFINER=‘root‘@‘localhost‘ FUNCTION  ‘bases‘.‘agregarseismeses‘(fecha DATETIME)
RETURNS datetime
BEGIN
return DATE_ADD(fecha, INTERVAL 6 MONTH);
```

```
END
```

4.7. Testing

El archivo “impl/test-data.sql” de la entrega, contiene instrucciones de SQL para probar las estructuras creadas.