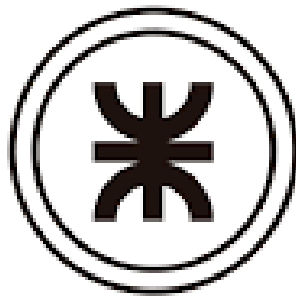


GRUPO: 20 - LOS DESNORMALIZADOS

Gestión de Datos
Segundo cuatrimestre 2021
Trabajo práctico - Gestión de Flota



UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

CURSO: K3521

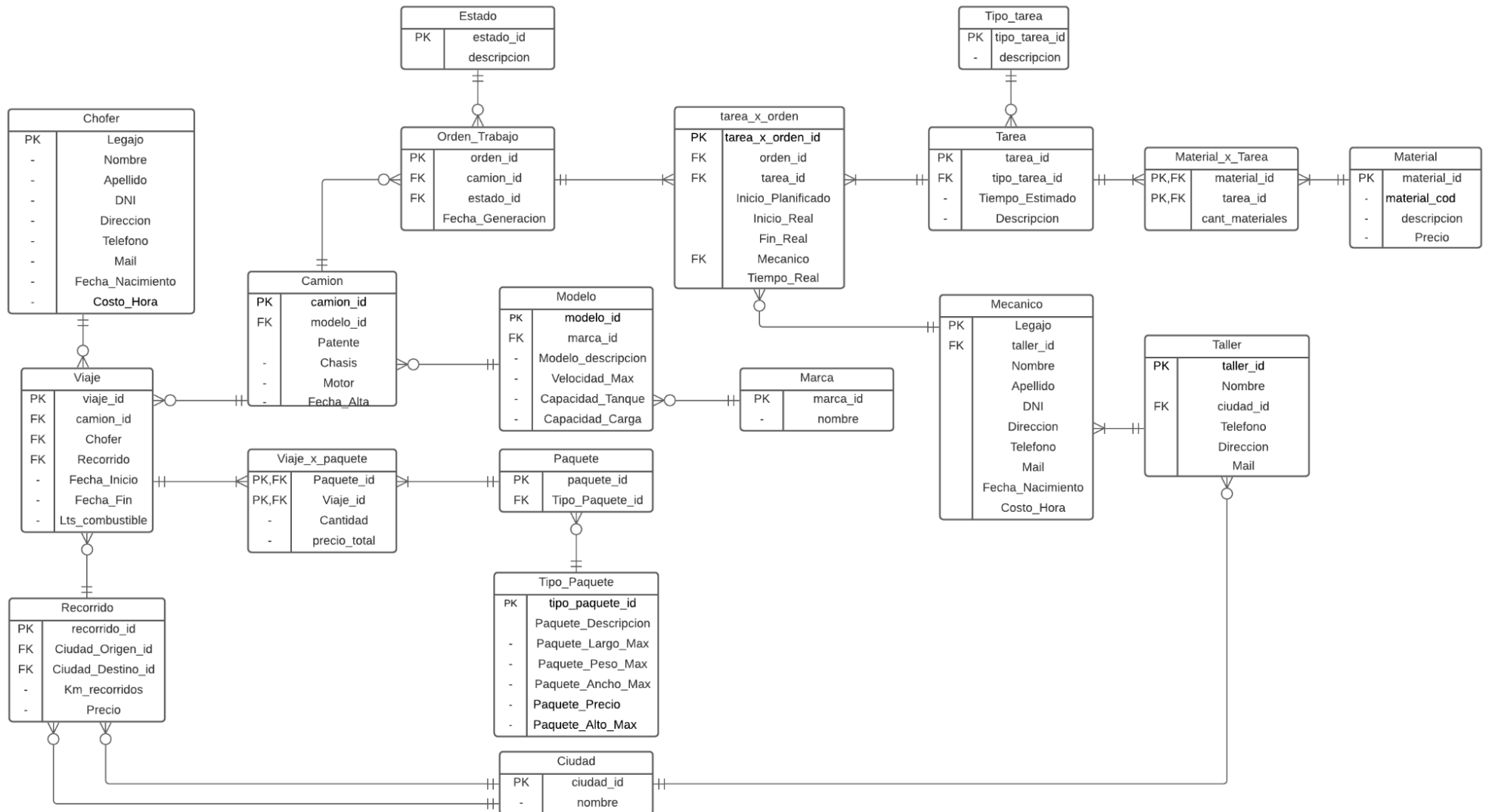
INTEGRANTES:

- **Blanco, Carolina** - 1715951
- **Villarreal, Agustín** - 1715525
- **Davicino, Matias** - 1715495
- **Guillan, Camila** -1723212

GRUPO: 20 - LOS DESNORMALIZADOS

DER.....	P.3
INTRODUCCIÓN.....	P.4
DOCUMENTACIÓN DER.....	P.4
DOCUMENTACIÓN MIGRACIÓN.....	P.8

GRUPO: 20 - LOS DESNORMALIZADOS



GRUPO: 20 - LOS DESNORMALIZADOS

INTRODUCCIÓN

En el presente documento se desarrollará la estrategia utilizada y las decisiones tomadas para el sistema de gestión de viajes y mantenimiento de camiones de una empresa que se encarga de transportar paquetes. Siguiendo el Diagrama de Entidad de Relación (DER), crearemos un modelo de datos para organizar los mismos, para más adelante migrarlos e implementarlos.

Documentación DER Gestión de Flota

Tras leer el enunciado, decidimos comenzar modelando “Viaje” ya que es la primera entidad de la consigna y consideramos que es la principal para nuestro modelo, la cual tiene diferentes FK, relacionándola con las tablas “Camion”, “Chofer” y “Recorrido”, que serán detalladas más adelante.

La clase “Viaje” tiene los siguientes atributos:

- VIAJE_ID (PK)
- CAMION_ID (FK => CAMION)
- CHOFER (FK => CHOFER)
- RECORRIDO (FK => RECORRIDO)
- FECHA_INICIO
- FECHA_FIN
- LTS_COMBUSTIBLE

Continuando con la lectura y como dijimos previamente, modelamos las tablas “Camion”, “Chofer” y “Recorrido”, teniendo en cuenta que el camión es el que realiza el viaje, el chofer maneja el camión y el viaje tiene un recorrido.

La clase “Camion” tiene los siguientes atributos:

- CAMION_ID (PK)
- MODELO_ID (FK => MODELO)
- PATENTE
- CHASIS
- MOTOR
- FECHA_ALTA

La clase “Chofer” tiene los siguientes atributos:

- LEGAJO (PK)
- NOMBRE
- APELLIDO
- DNI

GRUPO: 20 - LOS DESNORMALIZADOS

- DIRECCION
- TELEFONO
- MAIL
- FECHA_NACIMIENTO
- COSTO_HORA

La clase “Recorrido” tiene los siguientes atributos:

- RECORRIDO_ID (PK)
- CIUDAD_ORIGEN_ID (FK => CIUDAD)
- CIUDAD_DESTINO_ID (FK => CIUDAD)
- KM_RECORRIDOS
- PRECIO

Decidimos que las tablas “Recorrido” y “Camion” tengan su propio ID para facilitar el acceso a las mismas y evitar repeticiones. A su vez, creamos dos tablas, “Modelo” y “Marca”, las cuales nos permiten normalizar el modelo. La tabla Modelo tiene como FK el ID de la Marca.

La clase “Modelo” tiene los siguientes atributos:

- MODELO_ID (PK)
- MARCA_ID (FK => MARCA)
- MODELO_DESCRIPCION
- VELOCIDAD_MAX
- CAPACIDAD_TANQUE
- CAPACIDAD_CARGA

La clase “Marca” tiene los siguientes atributos:

- MARCA_ID (PK)
- NOMBRE

Ambas tienen como PK un ID específico (MODELO_ID y MARCA_ID), que permiten identificar al Modelo y a la Marca de forma unívoca, respectivamente.

Más adelante, modelamos la tabla “Paquete”. Decidimos implementar una tabla intermedia llamada Viaje_X_Paquete, ya que un paquete podría estar en muchos viajes y un viaje podría tener varios paquetes. Rompemos la relación de muchos a muchos con esta tabla, pudiendo así otorgarle la cantidad que hay de cada paquete en cada viaje.

GRUPO: 20 - LOS DESNORMALIZADOS

Por otro lado, creamos la tabla “Tipo_Paquete”, ya que hay 3 tipos de paquetes cuyas especificaciones están estandarizadas. Por viaje, se llevan varios paquetes de distinto tipo.

La clase “Paquete” tiene los siguientes atributos:

- PAQUETE_ID (PK)
- TIPO_PAQUETE_ID (FK => TIPO_PAQUETE_ID)

La clase “Tipo_Paquete” tiene los siguientes atributos:

- TIPO_PAQUETE_ID (PK)
- PAQUETE_DESCRIPCION
- PAQUETE_LARGO_MAX
- PAQUETE_PESO_MAX
- PAQUETE_ANCHO_MAX
- PAQUETE_PRECIO
- PAQUETE_ALTO_MAX

La clase “Viaje_x_paquete” tiene los siguientes atributos:

- PAQUETE_ID (PK, FK => PAQUETE)
- VIAJE_ID (PK, FK => VIAJE)
- CANTIDAD
- PRECIO_TOTAL

Seguimos modelando la tabla “Orden_Trabajo”. Esta también tiene su propio ID como PK y se relaciona con el ID del camión (FK). Además, decidimos hacer una tabla “Estado” que contenga el ID del estado y la descripción del mismo, permitiéndonos normalizar los datos.

La clase “Orden_Trabajo” tiene los siguientes atributos:

- ORDEN_ID (PK)
- CAMION_ID (FK => CAMION)
- ESTADO_ID (FK => ESTADO_ID)
- FECHA_GENERACION

La clase “Estado” tiene los siguientes atributos:

- ESTADO_ID (PK)
- DESCRIPCION

Siguiendo con la lectura, modelamos la tabla de “Tarea”. En este caso también decidimos romper la relación de muchos-muchos que se da con una

GRUPO: 20 - LOS DESNORMALIZADOS

tabla intermedia llamada “Tarea_X_Orden”, ya que en una orden puede haber muchas tareas, y una tarea puede estar en muchas órdenes. Utilizando esta tabla, podemos darle los atributos específicos que cada tarea tiene según la orden, como por ejemplo el inicio y fin real, el inicio planificado, el mecánico que la llevará a cabo y el tiempo real que tardó.

Además, usamos la tabla “Tipo_Tarea” que tiene un ID del tipo de la tarea (que es la FK de la tarea) y la descripción del tipo (preventivas o correctivas).

La clase “Tarea” tiene los siguientes atributos:

- TAREA_ID (PK)
- TIPO_TAREA_ID (FK => TIPO_TAREA)
- TIEMPO_ESTIMADO
- DESCRIPCION

La clase “Tipo_Tarea” tiene los siguientes atributos:

- TIPO_TAREA_ID (PK)
- DESCRIPCION

La clase “tarea_x_orden” tiene los siguientes atributos:

- TAREA_X_ORDEN_ID (PK)
- ORDEN_ID (FK => ORDEN_TRABAJO)
- TAREA_ID (FK => TAREA)
- INICIO PLANIFICADO
- INICIO_REAL
- FIN_REAL
- MECANICO (FK => MECANICO)
- TIEMPO_REAL

Acercándonos al final de la consigna, modelamos las clases “Material” y “Material_X_Tarea”. Esta última la utilizamos para romper la relacion muchos-muchos que se da entre material y tarea, ya que una tarea puede tener muchos materiales y un material puede estar en muchas tareas.

La clase “Material” tiene los siguientes atributos:

- MATERIAL_ID (PK)
- MATERIAL_COD
- DESCRIPCION
- PRECIO

GRUPO: 20 - LOS DESNORMALIZADOS

La clase “Material_x_Tarea” tiene los siguientes atributos:

- MATERIAL_ID (PK, FK => MATERIAL)
- TAREA_ID (PK, FK => TAREA)
- CANT_MATERIALES

Finalmente, creamos las tablas “Mecanico” y “Taller”, siguiendo lo planteado en la consigna. Al taller le agregamos un ID como PK para poder identificarlo unívocamente, al igual que hicimos en la tabla “Ciudad”.

La clase “Mecanico” tiene los siguientes atributos:

- LEGAJO (PK)
- TALLER_ID (FK => TALLER)
- NOMBRE
- APELLIDO
- DNI
- DIRECCION
- TELEFONO
- MAIL
- FECHA_NACIMIENTO
- COSTO_HORA

La clase “Taller” tiene los siguientes atributos:

- TALLER_ID (PK)
- NOMBRE
- CIUDAD_ID (FK => CIUDAD)
- TELEFONO
- DIRECCION
- MAIL

La clase “Ciudad” tiene los siguientes atributos:

- CIUDAD_ID (PK)
- NOMBRE

DOCUMENTACIÓN MIGRACIÓN

Comenzamos el proceso de la migración creando las tablas que vamos a usar y estableciendo los tipos de datos de los atributos de cada una de ellas, que son acordes a los de la tabla maestra. A modo de ejemplo, mostraremos cómo creamos la tabla de Mecánico.

GRUPO: 20 - LOS DESNORMALIZADOS

```
IF OBJECT_ID ('los_desnormalizados.Mecanico', 'U') IS NOT NULL
    DROP TABLE los_desnormalizados.Mecanico;
GO
CREATE TABLE los_desnormalizados.Mecanico(
    legajo INT PRIMARY KEY,
    nombre NVARCHAR(255) NOT NULL,
    apellido NVARCHAR(255) NOT NULL,
    dni DECIMAL(18,0) NOT NULL,
    direccion NVARCHAR(255) NOT NULL,
    telefono INT NOT NULL,
    mail NVARCHAR(255) NOT NULL,
    fecha_nacimiento DATETIME2(3) NOT NULL,
    costo_hora INT NOT NULL,
    taller_id INT NOT NULL
)
```

Para evitar cualquier tipo de problema a la hora de ejecutar la migración, creamos las FK al final alterando las tablas.

```
ALTER TABLE los_desnormalizados.Mecanico
ADD
    FOREIGN KEY (taller_id) REFERENCES los_desnormalizados.Taller (taller_id)
GO
```

Luego de esto, decidimos realizar los procedimientos para insertar los datos en las tablas creadas anteriormente.

En el presente documento se detallarán consideraciones pertinentes a algunas de las tablas de la migración puesto que los casos restantes son triviales.

Para la migración de la tabla “**Ciudad**”, los datos se encuentran repartidos en “**RECORRIDO_CIUADAD_DESTINO**”, “**RECORRIDO_CIUADAD_ORIGEN**”, “**TALLER_CIUADAD**” de la tabla Maestra, por lo tanto decidimos resolverlo con un UNION como se observa en la siguiente query.

GRUPO: 20 - LOS DESNORMALIZADOS

```
195 -- Ciudad
196 CREATE PROCEDURE los_desnormalizados.migracionCiudad
197 as
198 begin
199 INSERT INTO los_desnormalizados.Ciudad (nombre)
200     SELECT DISTINCT RECORRIDO_CIUADAD_DESTINO FROM gd_esquema.Maestra
201     WHERE RECORRIDO_CIUADAD_DESTINO IS NOT NULL
202 UNION
203     SELECT DISTINCT RECORRIDO_CIUADAD_ORIGEN FROM gd_esquema.Maestra
204     WHERE RECORRIDO_CIUADAD_ORIGEN IS NOT NULL
205 UNION
206     SELECT DISTINCT TALLER_CIUADAD FROM gd_esquema.Maestra
207     WHERE TALLER_CIUADAD IS NOT NULL
208 END
```

Para la tabla “Material_x_tarea”, decidimos calcular el promedio de la cantidad de materiales para cada tarea existente en una orden dentro de un subquery. Esto se debe a que una tarea podía ser realizada por diferentes materiales dependiendo de la orden en la que se encuentre.

```
-- Material_x_tarea
CREATE PROCEDURE los_desnormalizados.migracionMaterialXTarea
as
begin
INSERT INTO los_desnormalizados.Material_x_tarea (material_id, tarea_id, cant_material)
    SELECT DISTINCT material_id,
                    tarea_id,
                    COUNT(m.MATERIAL_COD) / (select count(distinct convert(varchar,TAREA_FECHA_FIN)
                    +convert(varchar, TAREA_FECHA_INICIO)
                    +convert(varchar, TAREA_FECHA_INICIO_PLANIFICADO)
                    +str(TAREA_TIEMPO_ESTIMADO)+ CAMION_PATENTE)
                    FROM gd_esquema.Maestra
                    WHERE TAREA_CODIGO= m.TAREA_CODIGO)
    FROM gd_esquema.Maestra m
    JOIN los_desnormalizados.Material mate ON (m.MATERIAL_DESCRIPCION = mate.material_descripcion)
    JOIN los_desnormalizados.Tarea t ON (t.tarea_id = m.TAREA_CODIGO)
    GROUP BY TAREA_CODIGO, m.MATERIAL_COD, m.MATERIAL_DESCRIPCION, material_id, tarea_id
END
```

Para la tabla “Tarea_x_orden”, decidimos crear una PK IDENTITY(1,1). En un primer momento se planteó que “tarea_id” y “orden_id” sean PK pero debido a que más de un mecánico podría realizar una misma tarea dentro de una misma orden se decidió crear un índice autoincremental, quedando tarea_id y orden_id como FK. Además agregamos el tiempo_real utilizando una función que nos provee el motor de SQL Server que hace la diferencia entre “fin_real” e “inicio_real”.

GRUPO: 20 - LOS DESNORMALIZADOS

```
382 -- Tarea_x_orden
383 CREATE PROCEDURE los_desnormalizados.migracionTareaXorden
384 as
385 begin
386 INSERT INTO los_desnormalizados.Tarea_x_orden (orden_id, tarea_id, mecanico_id, inicio_planificado, inicio_real,
387 fin_real, tiempo_real)
388 SELECT DISTINCT orden_id, tarea_id, legajo, TAREA_FECHA_INICIO_PLANIFICADO, TAREA_FECHA_INICIO, TAREA_FECHA_FIN, DATEDIFF(day, TAREA_FECHA_INICIO, TAREA_FECHA_FIN)
389 FROM gd_esquema.Maestra m
390 JOIN los_desnormalizados.Camion c ON (c.patente = m.CAMION_PATENTE)
391 JOIN los_desnormalizados.Orden_trabajo ot ON (ot.fecha_generacion = m.ORDEN_TRABAJO_FECHA AND
392 ot.camion_id = c.camion_id)
393 JOIN los_desnormalizados.Tarea t ON (t.tarea_id = m.TAREA_CODIGO)
394 JOIN los_desnormalizados.Mecanico mec ON (mec.legajo = m.MECANICO_NRO_LEGajo)
395 END
```

Para la tabla “Viaje_x_paquete”, calculamos el precioTotal de todos los paquetes de un mismo tipo, ya que multiplicamos la cantidad de paquetes por el precio y lo sumamos al precio base del recorrido. Como se observa en la siguiente query.

```
334 -- Viaje_x_paquete
335 CREATE PROCEDURE los_desnormalizados.migracionViajeXPaquete
336 as
337 begin
338 INSERT INTO los_desnormalizados.Viaje_x_paquete (paquete_id, viaje_id, cantidad, precioTotal)
339 SELECT DISTINCT paquete_id, viaje_id, SUM(PAQUETE_CANTIDAD), SUM(PAQUETE_CANTIDAD) * tp.paquete_precio + r.precio
340 FROM gd_esquema.Maestra m
341 JOIN los_desnormalizados.Camion c ON (m.CAMION_PATENTE = c.patente)
342 JOIN los_desnormalizados.Viaje v ON (v.fecha_inicio = m.VIAJE_FECHA_INICIO AND
343 v.camion_id = c.camion_id)
344 JOIN los_desnormalizados.Tipo_paquete tp ON (tp.paquete_descripcion = m.PAQUETE_DESCRIPCION)
345 JOIN los_desnormalizados.Paquete p ON (p.tipo_paquete_id = tp.tipo_paquete_id)
346 JOIN los_desnormalizados.Recorrido r ON (r.recorrido_id = v.recorrido_id)
347 group by viaje_id, paquete_id, tp.paquete_precio, r.precio
348 END
```

A modo de ejemplo, se mostrará un query de la migración de los datos de la tabla mecánico teniendo en cuenta que el resto de las tablas se realizan de forma similar.

```
369 --Mecanico
370 CREATE PROCEDURE los_desnormalizados.migracionMecanico
371 as
372 begin
373 INSERT INTO los_desnormalizados.Mecanico (legajo, nombre, apellido, dni, direccion, telefono, mail,
374 fecha_nacimiento, costo_hora, taller_id)
375 SELECT DISTINCT MECANICO_NRO_LEGajo, MECANICO_NOMBRE, MECANICO_APELLIDO, MECANICO_DNI,
376 MECANICO_DIRECCION, MECANICO_TELEFONO, MECANICO_MAIL, MECANICO_FECHA_NAC, MECANICO_COSTO_HORA,
377 t.taller_id FROM gd_esquema.Maestra m
378 JOIN los_desnormalizados.taller t ON (t.nombre = TALLER_NOMBRE)
379 WHERE MECANICO_NRO_LEGajo IS NOT NULL
380 END
```