P2 Arquitectura de Datos

UC3M



**David Merino Hernández 100432028**

**Ángela Morales Valverde 100432200**

**Samuel Fernández Fernández 100432070**

**ÍNDICE**

[1. INTRODUCCIÓN 3](#_yspy8tt3f0xe)

[2. MODELADO DE LA INFORMACIÓN 3](#_75rf4vta81ax)

[3. DIAGRAMA DE CLASES CASSANDRA 4](#_6nqf6khj3ynz)

[4. CASOS DE USO 6](#_athwf8tpi37f)

[4.1 Estudios por marcas y modelos (CU\_A) 6](#_3wf1t5m95m2w)

[4.1.2 Número de multas por marca y modelo de vehículo (CU\_A1) 6](#_zfl9ea3hwro1)

[4.1.3 Los tres colores de vehículos más multados (CU\_A2) 6](#_v2h3hkuj0tt9)

[4.1.1 Marcas y modelos cuyos conductores respetan menos los límites de velocidad (CU\_A3) 6](#_x4apgp8eolr8)

[4.2 Estudio por carreteras (CU\_B) 6](#_gevg8tsw2cq6)

[4.2.1 Exceso de velocidad medio en porcentaje (CU\_B1) 6](#_uks3091y3ekw)

[4.2.2 Tramo y sentido de carretera más conflictivo (CU\_B2) 7](#_o8xoirmsv33q)

[4.3 Estudio por conductores (CU\_C) 7](#_x82xlg2upsjc)

[4.3.1 Conductores más infractores (CU\_C1) 7](#_itffwa8v0uy)

[4.3.2 Probabilidad de infracción cuando un vehículo no lo conduce su dueño (CU\_C2) 7](#_7wyuuuxbqswx)

[4.4 Creación y gestión de expedientes de sanciones (CU\_D) 7](#_eld7rfn8reyt)

[4.5 Gestión de expedientes activos para proceso ejecutivo (CU\_E) 8](#_79pk6mcossvm)

[5. CONSULTAS 8](#_lyh6hh7t4cgs)

[5.1 Número de multas por marca y modelo de vehículo 8](#_k86brg6bnvi4)

[5.2 Los tres colores de vehículos más multados 9](#_5j4hztmilm2v)

[5.3 Marcas y modelos cuyos conductores respetan menos los límites de velocidad 10](#_yyyad1ckk8v1)

[5.4 Exceso de velocidad medio en porcentaje 11](#_z0yxowaqju6w)

[5.5 Tramo y sentido de carretera más conflictivo 11](#_melviuqf3fwi)

[5.6 Conductores más infractores 12](#_o2zn9lmytslo)

[5.7 Probabilidad de infracción cuando un vehículo no lo conduce su dueño 13](#_f40sdzgfxrrj)

[5.8 Creación y gestión de expedientes de sanciones 13](#_k9pv2d9zqnmh)

[5.9 Gestión de expedientes activos para proceso ejecutivo 14](#_i85htxc6f9gv)

[6. TABLAS 15](#_e2aw1k39v9ya)

[6.1 Tabla radar 15](#_ldk1u3nu3hc)

[6.2 Tabla vehicle 16](#_e8idvsppqwjw)

[6.3 Tabla person 16](#_bvz92yvkme3v)

[6.4 Tabla violation 17](#_fxzzx1iuxvbq)

[6.5 Tabla speed\_ticket 18](#_idtetfx0y3y2)

[6.6 Tabla roadworthiness 18](#_i1k44bwaqi8k)

[6.7 Tabla record 19](#_o6tc2qls4h0k)

[6.8 Tabla vehicle\_by\_make\_model 19](#_6a2x2fatcadr)

[6.9 Tabla colour\_violation\_count 20](#_gjs7p5nu6j84)

[6.10 Tabla speed\_excess\_avg 20](#_w3ngaqy25xh0)

[6.11 Tabla avg\_speed\_excess\_by\_by\_mileage\_direction 21](#_efno9rigugbn)

[6.12 Tabla driver\_violation\_count 21](#_ol8zobmhkixp)

[6.13 Tabla non\_owner\_violations 22](#_snlr71l4z76q)

[6.14 Tabla active\_violations 22](#_cb3kues2b63)

[6.15 Tabla speed\_excess\_by\_road 23](#_mpws74fiktd4)

[7. PROCESO DE MIGRACIÓN Y CARGA DE DATOS 23](#_it68fwh5skm8)

[8. RESULTADOS OBTENIDOS 25](#_kfuxf8z11b7m)

[9. CONCLUSIONES 25](#_l66p1qybyhsl)

[10. ANEXOS 26](#_i14hq8cowfrl)

[11.1 Tablas iniciales de MongoDB 26](#_k5dfvgz7zq6q)

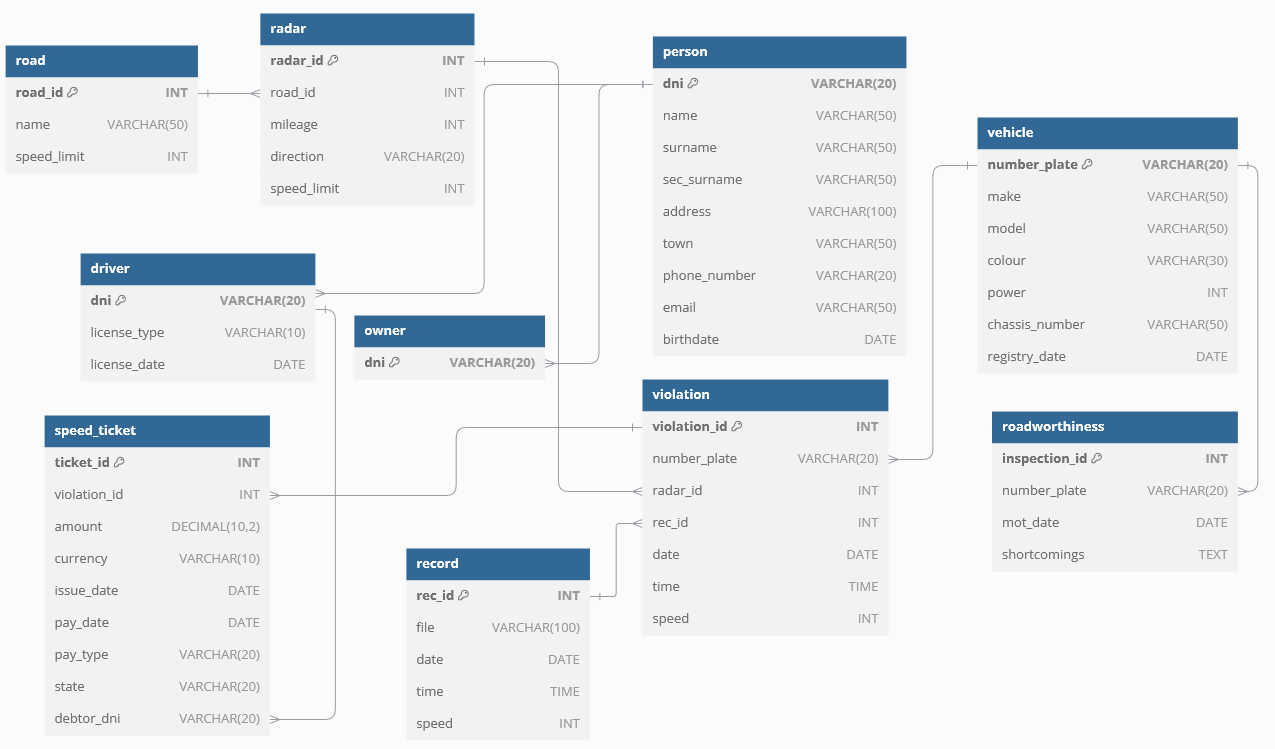
# 

# **1. INTRODUCCIÓN**

En esta práctica se nos pide adaptar y migrar los datos relacionados con las sanciones por velocidad impuestas por la DGT desde una estructura de MongoDB a una nueva estructura basada en Apache Cassandra. Para ello, se tendrá que hacer un estudio de los datos existentes en el dataset que se nos ha proporcionado para posteriormente adaptarlos a la estructura de consultas y tablas que tendremos que diseñar para el nuevo modelo en Cassandra.

# **2. MODELADO DE LA INFORMACIÓN**

Una posible representación de esta estructura de datos que recibimos desde MongoDB podría ser la que se ve en la siguiente imagen, donde se pueden apreciar las distintas relaciones y las claves para cada una de las tablas.



De la clase “person”, con clave primaria el DNI, se puede observar que derivan las clases de “driver” y “owner”. Así como se pueden ver el resto de relaciones que están presentes entre las distintas tablas que se reciben.

Para completar estas relaciones y el modelado destacamos que:

* Que un expediente tenga una sanción (*speed\_ticket)* es opcional (0..1).
* Un vehículo (*vehicle*) puede tener una o más Revisiones técnicas (*roadworthiness*) (1..\*).
* Dentro de una sanción (*speed\_ticket*) el tipo de pago (*pay\_type*) y la fecha de pago (*pay\_date*) son opcionales.
* El número de teléfono (*phone\_number*) y el email ambos dentro de persona (*person*) son opcionales.
* Un propietario (*owner*) es una persona (*person*).
* El resto de relaciones son 1:1

# **3. DIAGRAMA DE CLASES CASSANDRA**

Una vez analizados los datos que se tienen, se estructura una serie de clases en las que se van a tener que adaptar los datos a la nueva estructura necesaria para poder trabajar con Cassandra.

Como se puede observar, no hay relaciones entre las tablas, ya que estas almacenan en su totalidad la información que es necesaria para cada uno de los casos de uso que se explicarán más adelante en el trabajo. También, se han adaptado las claves primarias y los atributos de dichas clases para que sean coherentes con la información que se va a necesitar obtener posteriormente en las consultas que se realicen.

Esta aproximación de las tablas se fundamenta en los casos de uso y relaciones que se van a explicar en profundidad más adelante en el trabajo y son resultado de crear las tablas conforme a la información que se necesita.



# **4. CASOS DE USO**

## **4.1 Estudios por marcas y modelos (CU\_A)**

Estos casos de uso que vienen a continuación se centran en datos que se quieren obtener sobre los vehículos para así poder usarlos posteriormente en el análisis estadístico.

### 4.1.2 Número de multas por marca y modelo de vehículo (CU\_A1)

En este subcaso de uso se trata de averiguar la cantidad de multas que son emitidas por cada marca y modelo de vehículo, por lo que se deberá disponer de una tabla que recoja esta información.

### 4.1.3 Los tres colores de vehículos más multados (CU\_A2)

En este subcaso de uso se pretende poder identificar y ordenar el color de los vehículos que más multas tienen registradas, por lo que se deberá crear una tabla que tenga esta información y que la ordene para poder mostrar exáctamente lo que se pide.

### 4.1.1 Marcas y modelos cuyos conductores respetan menos los límites de velocidad (CU\_A3)

En este subcaso de uso se quiere evaluar de entre todos los vehículos que se tienen cuál es el exceso promedio de velocidad que tienen, y evaluarlo según la marca y el modelo del vehículo. De igual manera, se tendrá que crear una tabla exclusiva que esté dedicada a mostrar esta información.

## **4.2 Estudio por carreteras (CU\_B)**

Estos casos de uso que vienen a continuación se centran en datos que se quieren obtener sobre las carreteras para así poder usarlos posteriormente en el análisis estadístico.

### 4.2.1 Exceso de velocidad medio en porcentaje (CU\_B1)

Centrándose en las carreteras esta vez, se quiere saber el exceso medio de velocidad en cada una de esas carreteras, creando una tabla que almacene esta información para luego poder consultar dicha información de alguna carretera en específico.

### 4.2.2 Tramo y sentido de carretera más conflictivo (CU\_B2)

Para cada una de las carreteras, se quiere saber el tramo en kilómetros y sentido de circulación que tiene un mayor índice de exceso de velocidad, creando una tabla con la información de las carreteras que luego se pueda consultar.

## **4.3 Estudio por conductores (CU\_C)**

Estos casos de uso que vienen a continuación se centran en datos que se quieren obtener sobre los conductores para así poder usarlos posteriormente en el análisis estadístico.

### 4.3.1 Conductores más infractores (CU\_C1)

Se quiere saber cuáles son los conductores que más infracciones tienen registradas, buscando obtener el número exacto.

### 4.3.2 Probabilidad de infracción cuando un vehículo no lo conduce su dueño (CU\_C2)

Aparte de los propietarios de los vehículos, de los que son conductores no propietarios, se busca poder obtener información sobre la probabilidad de que éstos hagan una infracción.

## **4.4 Creación y gestión de expedientes de sanciones (CU\_D)**

De forma implícita en el enunciado se dice que se quiere obtener información para poder crear expedientes de sanciones desde los datos recogidos por los radares y los agentes de tráfico. Para ello, se busca analizar las pruebas, evaluar los factores relevantes para la o las infracciones y emitirlas en caso de que existan.

Concretamente, se deberán obtener las pruebas, determinar si el exceso de velocidad que se haya visto es sancionable o no, registrar los datos del conductor y vehículo que están involucrados en la sanción, emitir las sanciones necesarias en caso que exista más de una y analizar el estado de los expedientes que existan.

## **4.5 Gestión de expedientes activos para proceso ejecutivo (CU\_E)**

De igual manera que el anterior, se dice implícitamente en el enunciado que se pretende poder identificar los expedientes que no se hayan cerrado en el plazo normal de pago para así poder enviarlos al proceso ejecutivo pertinente.

Para ello, será necesario registrar los expedientes existentes que cumplan estas características y mantener un registro de las fechas de emisión y límite de pago de las sanciones.

# **5. CONSULTAS**

Una vez que se hizo el diagrama de cómo debería quedar el diseño para usarse en Cassandra, se deben crear las consultas que permitan obtener la información que se necesita en cada uno de los casos de uso. Para ello, se ha creado una consulta específica para cada uno de ellos junto el establecimiento de la clave de partición y la de clusterización en la tabla objetivo, además el código necesario para la creación de dichas tablas se establece en el punto 6.

## **5.1 Número de multas por marca y modelo de vehículo**

Esta consulta hace referencia al caso de uso CU\_A1 y tendría una estructura como la que se muestra a continuación.

Esta consulta debería poder ordenar los resultados obtenidos de forma general según el número de multas como se muestra en la primera consulta, pero se ha comprobado que puesto de dicha forma, y por la estructura que tiene Cassandra, esto nos conduce a un error. Al no tener una cláusula WHERE no nos permite hacer una ordenación de todos los datos.

| SELECT make, model, COUNT(\*) AS ticket\_count  FROM vehicle\_by\_make\_model  GROUP BY make, model  ORDER BY ticket\_count; |
| --- |

**Tabla:** vehicle\_by\_make\_model

En cambio, si se intenta poner la consulta sin el criterio de ordenación, para poder ver así qué datos se obtienen, vemos que sí que se consigue saber cuantas multas tiene cada vehículo según su marca y su modelo.

| SELECT make, model, COUNT(\*) AS ticket\_count  FROM vehicle\_by\_make\_model  GROUP BY make, model; |
| --- |

**Tabla:** vehicle\_by\_make\_model

De esta forma, a través de la tabla “**vehicle\_by\_make\_model**”, se recogerá la información que se requiere en esta consulta. Se almacena la información de los vehículos junto con el conteo de multas, para después agruparse por marca y modelo para así obtener el número total de multas.

## **5.2 Los tres colores de vehículos más multados**

Esta consulta hace referencia al caso de uso CU\_A2 y tendría una estructura como la que se muestra a continuación.

De igual manera que con la consulta anterior, al tratar de establecer un orden total de los datos obtenidos sin una cláusula WHERE obtenemos un error por parte de Cassandra. La consulta que sería exactamente como el caso de uso lo requiere sería la siguiente.

| SELECT colour, violation\_count  FROM colour\_violation\_count  ORDER BY violation\_count  LIMIT 3; |
| --- |

**Tabla:** violations\_by\_colour

Para poder ver los datos que se obtienen aunque no estén ordenados se quita el ordenamiento y sí que se pueden ver los colores de los vehículos según el número de infracciones que tengan registradas.

| SELECT colour, violation\_count  FROM colour\_violation\_count  LIMIT 3; |
| --- |

**Tabla:** violations\_by\_colour

De esta forma, a través de la tabla “**colour\_violation\_count**”, se pueden contar las infracciones agrupadas según el color del vehículo y ordenarlas después de mayor a menor para así obtener los tres que son más multados.

## **5.3 Marcas y modelos cuyos conductores respetan menos los límites de velocidad**

Esta consulta hace referencia al caso de uso CU\_A3 y tendría una estructura como la que se muestra a continuación.

Como en este caso también que debería ordenar los resultados obtenidos de la consulta según el exceso medio en promedio de velocidad, vuelve a ocurrirnos lo mismo de que nos aparezca el error de Cassandra al carecer de la cláusula WHERE. Técnicamente la consulta que se ajusta al caso de uso que se ha nombrado es la que se ve a continuación.

| SELECT make, model, avg\_speed\_excess  FROM speed\_excess\_avg  ORDER BY avg\_speed\_excess DESC; |
| --- |

**Tabla:** speed\_excess\_by\_road

Como esto no nos proporcionaba una visión de los datos, quitamos la ordenación de la consulta para ver los datos, aunque no estuvieran ordenados, y sí que se consiguen los datos que se necesitan para poder cubrir el caso de uso.

| SELECT make, model, avg\_speed\_excess  FROM speed\_excess\_avg; |
| --- |

**Tabla:** speed\_excess\_by\_road

De esta forma, a través de la tabla “**speed\_excess\_avg**”, se puede obtener el valor del exceso de velocidad promedio por marca y modelo para después almacenar dicho exceso.

## **5.4 Exceso de velocidad medio en porcentaje**

Esta consulta hace referencia al caso de uso CU\_B1 y tendría una estructura como la que se muestra a continuación.

Al ir buscando obtener un único valor, sí que obtenemos el dato que se necesita.

| SELECT AVG((speed - speed\_limit) \* 100 / speed\_limit) AS avg\_excess\_percentage  FROM speed\_excess\_by\_road  WHERE road\_name = 'M30'; |
| --- |

**Tabla:** speed\_excess\_by\_road

De esta forma, a través de la tabla “**speed\_excess\_by\_road**”, se puede obtener el porcentaje de exceso de velocidad promedio para una carretera específica. Vemos que si ejecutamos esta consulta nos devuelve un valor que equivaldría a dicho porcentaje de exceso de velocidad.

## **5.5 Tramo y sentido de carretera más conflictivo**

Esta consulta hace referencia al caso de uso CU\_B2 y tendría una estructura como la que se muestra a continuación.

En este caso, como se quiere ordenar la información con la cláusula ORDER BY pero en este caso sí que tenemos una cláusula WHERE, no nos da el error de Cassandra y sí que obtenemos la información sobre la carretera que se seleccione con los datos ordenados según el criterio que hemos establecido.

| SELECT mileage, direction, avg\_speed\_excess  FROM avg\_speed\_excess\_by\_mileage\_direction  WHERE road\_name = 'M30'  ORDER BY avg\_speed\_excess DESC  LIMIT 10; |
| --- |

**Tabla:** violations\_by\_mileage\_direction

De esta forma, a través de la tabla “**avg\_speed\_excess\_by\_mileage\_direction**”, se puede conseguir obtener el tramo y el sentido de una carretera específica con mayor exceso de velocidad promedio. Al ejecutar esa consulta nos sale ordenado de mayor a menor según el exceso medio de velocidad los tramos y los sentidos de una cierta carretera.

## **5.6 Conductores más infractores**

Esta consulta hace referencia al caso de uso CU\_C1 y tendría una estructura como la que se muestra a continuación.

Como en casos anteriores, no podemos ordenar los datos globales obtenidos de la columna porque Cassandra no lo permite, aunque sería lo que cumpliría estrictamente el caso de uso al que representa.

| SELECT driver\_dni, violation\_count  FROM driver\_violation\_count  ORDER BY violation\_count  LIMIT 10; |
| --- |

**Tabla:** driver\_violations

Al quitar esta cláusula de ordenación sí que obtenemos datos representantes de los conductores con sus respectivas infracciones.

| SELECT driver\_dni, violation\_count  FROM driver\_violation\_count  LIMIT 10; |
| --- |

**Tabla:** driver\_violations

De esta forma, a través de la tabla “**driver\_violation\_count**”, en la cual se almacenan todas las infracciones cometidas por cada conductor, ordenándose por el número de infracciones de emayor a menor, pudiendo así obtener los conductores con más infracciones.

## **5.7 Probabilidad de infracción cuando un vehículo no lo conduce su dueño**

Esta consulta hace referencia al caso de uso CU\_C2 y tendría una estructura como la que se muestra a continuación.

En este caso para poder obtener la información hay que hacerlo en dos partes que se muestran a continuación.

| SELECT COUNT(\*) AS non\_owner\_violations\_count  FROM non\_owner\_violations;  SELECT COUNT(\*) AS total\_violations\_count  FROM violation; |
| --- |

**Tablas:** non\_owner\_violations\_count y violation

De esta forma, a través de la tabla “**non\_owner\_violations\_count**”, se puede contar el número de infracciones cometidas por conductores que no son los dueños de los vehículos y se compara con el total de infracciones cometidas para calcular la probabilidad.

Como se puede observar, se ha conseguido la información de dos consultas distintas, dejando así para cálculo externo la probabilidad de infracción, ya que dentro de Cassandra no hay una forma en la que podamos obtener este dato dentro de tan solo una consulta.

La probabilidad se calcula con una herramienta externa, bien sea con un programa de Python u otra, y sigue la siguiente fórmula: probabilidad = (non\_owner\_violations\_count / total\_violations\_count) \* 100.

## **5.8 Creación y gestión de expedientes de sanciones**

Esta consulta hace referencia al caso de uso CU\_D y tendría una estructura como la que se muestra a continuación.

Para poder hacer una consulta ajustada a ese caso de uso se hace de la siguiente forma.

| SELECT violation\_id, status, issue\_date, due\_date  FROM active\_violations  WHERE status = 'active' AND due\_date < dateof(now()) ALLOW FILTERING; |
| --- |

**Tabla:** active\_violations

De esta forma, a través de la tabla “**active\_violations**”, se pueden listar los expedientes que aún están activos y cuyo plazo de pago ha vencido, indicando que necesitan ser enviados al proceso ejecutivo.

Al ejecutar esa consulta no obtenemos resultados y esto se debe a que en los datos que se obtuvieron la columna que hace referencia a “due date” aparece vacía en todos los casos. Se pone que se puede hacer ALLOW FILTERING para que pueda hacer la operación que le estamos diciendo que haga con las fechas.

## **5.9 Gestión de expedientes activos para proceso ejecutivo**

Esta consulta hace referencia al caso de uso CU\_E y tendría una estructura como la que se muestra a continuación.

La consulta quedaría como se puede ver más adelante, obteniendo información de tres sitios distintos, y sustituyendo en los sitios donde se encuentra “<x>” por los respectivos valores que se quieran buscar.

| SELECT \*  FROM violation;  WHERE violation\_id = <violation\_id>;  SELECT \*  FROM record  WHERE rec\_id = <rec\_id>;  SELECT \*  FROM speed\_ticket  WHERE violation\_id = <violation\_id>; |
| --- |

Para poder obtener la información que se pide en este caso, son necesarias las consultas que se han mostrado para así obtener todos los detalles relevantes del expediente, incluyendo la información sobre la infracción, el registro de la evidencia que exista sobre la infracción y los datos de la multa.

# **6. TABLAS**

Para estas consultas, hay que crear las tablas específicas diseñadas para cada caso de uso, así como las tablas generales que marcan los elementos del diseño inicial que se recibe desde MongoDB.

## **6.1 Tabla radar**

Las infracciones se recogen a través de una serie de radares, los cuales tienen una información asociada como un identificador e información sobre dónde están situados, y eso es lo que se recoge en la siguiente tabla.

| CREATE TABLE radar (  radar\_id TEXT PRIMARY KEY,  road\_name TEXT,  mileage INT,  direction TEXT,  speed\_limit INT  ); |
| --- |

La clave de particionado es el identificador del radar ya que es único para cada radar.

## **6.2 Tabla vehicle**

Las infracciones son cometidas por conductores en vehículos, por lo que es importante almacenar la información de dichos vehículos, ya sean datos técnicos de este o datos sobre la persona propietaria del vehículo.

| CREATE TABLE vehicle (  number\_plate TEXT PRIMARY KEY,  make TEXT,  model TEXT,  colour TEXT,  power INT,  chassis\_number TEXT,  registry\_date DATE,  owner\_dni TEXT  ); |
| --- |

La clave de partición para esta tabla es el número de matrícula del vehículo ya que es único para cada vehículo.

## **6.3 Tabla person**

Cada persona implicada en una infracción, sea conductor o no conductor, tiene una información asociada que se recoge en la siguiente tabla.

| CREATE TABLE person (  dni TEXT PRIMARY KEY,  name TEXT,  surname TEXT,  sec\_surname TEXT,  address TEXT,  town TEXT,  phone\_number TEXT,  email TEXT,  birthdate DATE  ); |
| --- |

La clave de partición es el identificador de la persona, que es el DNI, que es único para cada persona.

## **6.4 Tabla violation**

También se necesita poder almacenar la información sobre las infracciones que hay que identificarlas con un id, así como recoger la información sobre el vehículo y el conductor implicados en esta. También se almacenan datos relevantes sobre la infracción.

| CREATE TABLE violation (  violation\_id UUID PRIMARY KEY,  number\_plate TEXT,  radar\_id INT,  rec\_id UUID,  date DATE,  time TIME,  speed INT,  driver\_dni TEXT,  owner\_dni TEXT,  road\_name TEXT,  speed\_limit INT,  is\_infractor BOOLEAN,  status TEXT  ); |
| --- |

La clave de partición se decide que sea el identificador de la infracción ya que es único para cada infracción.

## **6.5 Tabla speed\_ticket**

Para el caso de almacenar la información sobre las multas por velocidad se crea una tabla que guarda el identificador de la infracción así como el de la multa. Asociado a ello se guardan datos sobre la sanción y sobre el conductor. Cada multa que se crea es única.

| CREATE TABLE speed\_ticket (  ticket\_id UUID PRIMARY KEY,  violation\_id UUID,  amount DECIMAL,  currency TEXT,  issue\_date DATE,  pay\_date DATE,  pay\_type TEXT,  state TEXT,  debtor\_dni TEXT  ); |
| --- |

La clave de partición para las multas de velocidad es el identificador del ticket de la sanción.

## **6.6 Tabla roadworthiness**

Para recoger las inspecciones que se hagan se ha creado la siguiente tabla que recoge la información pertinente que se recibe en la migración de datos.

| CREATE TABLE roadworthiness (  inspection\_id UUID PRIMARY KEY,  number\_plate TEXT,  mot\_date DATE,  shortcomings TEXT  ); |
| --- |

La clave de partición es el identificador de la inspección.

## **6.7 Tabla record**

Para esta tabla se han creado los atributos que se ven a continuación para recoger así la información que se va a recibir de la migración sobre los registros recogidos y almacenarla en una tabla concreta que se adapte al diseño que se recibe.

| CREATE TABLE record (  rec\_id UUID PRIMARY KEY,  file TEXT,  date DATE,  time TIME,  speed INT,  radar\_id INT  ); |
| --- |

En este caso la clave de partición es el identificador de este registro.

## **6.8 Tabla vehicle\_by\_make\_model**

Para poder llevar un conteo de infracciones según la marca y el modelo se crea la siguiente tabla en la que se registran ambas cosas, así como el número de matrícula del vehículo y el número total de multas que tiene dicho vehículo.

| CREATE TABLE vehicle\_by\_make\_model (  make TEXT,  model TEXT,  number\_plate TEXT,  ticket\_count INT,  PRIMARY KEY ((make, model), number\_plate)  ); |
| --- |

Para esta tabla la clave de partición se ha elegido que sea el modelo y la marca y que en el caso de la clave de clustering sea el número de matrícula.

## **6.9 Tabla colour\_violation\_count**

Para saber las infracciones que son registradas por cada color de vehículo se usa la siguiente tabla en la que se registra el color del vehículo y el identificador de la infracción.

| CREATE TABLE colour\_violation\_count (  colour text,  violation\_count int,  PRIMARY KEY (colour, violation\_count)  ) WITH CLUSTERING ORDER BY (violation\_count DESC); |
| --- |

Se ha elegido como clave de partición el color del vehículo y como clave de clustering el identificador de la infracción.

## **6.10 Tabla speed\_excess\_avg**

Para poder calcular el exceso de velocidad promedio para cada marca y modelo de vehículo se necesita la siguiente tabla en la que se recoge el identificador de la marca, el modelo y la velocidad de exceso media.

| CREATE TABLE speed\_excess\_avg (  make text,  model text,  avg\_speed\_excess double,  PRIMARY KEY (make, model, avg\_speed\_excess)  ) WITH CLUSTERING ORDER BY (model ASC, avg\_speed\_excess ASC); |
| --- |

Para hacer la partición se han elegido los tres atributos y como clave de clustering se ha elegido el modelo y la velocidad en exceso promedio

## **6.11 Tabla avg\_speed\_excess\_by\_by\_mileage\_direction**

Con la siguiente tabla se pretende poder almacenar la información sobre el tramo y el sentido más conflictivo por carretera. Para tener esta información se necesita saber el identificador de la infracción así como información de la carretera como el nombre de esta, el kilometraje y la dirección.

| CREATE TABLE avg\_speed\_excess\_by\_mileage\_direction (  road\_name text,  avg\_speed\_excess double,  mileage int,  direction text,  PRIMARY KEY (road\_name, avg\_speed\_excess, mileage, direction)  ) WITH CLUSTERING ORDER BY (avg\_speed\_excess ASC, mileage ASC, direction ASC); |
| --- |

Para esta tabla se han elegido como claveles de partición, el nombre de la carretera, el kilometraje y la dirección. Para el caso de la clave de clustering se ha decidido que sea el identificador de la infracción.

## **6.12 Tabla driver\_violation\_count**

Esta tabla ayuda a poder obtener las infracciones que son cometidas por cada conductor ordenadas por fecha. Para ello se necesita tener el identificador del conductor y el identificador de la misma.

| CREATE TABLE driver\_violation\_count (  driver\_dni text PRIMARY KEY,  violation\_count int  ); |
| --- |

En este caso se elige como clave de partición el identificador del conductor.

## **6.13 Tabla non\_owner\_violations**

Esta tabla permite almacenar las infracciones que son cometidas por conductores que no son los propietarios del vehículo que conducen. Para ello, es relevante almacenar la matrícula del vehículo, el identificador de la infracción, el identificador del propietario y el identificador del conductor.

| CREATE TABLE non\_owner\_violations (  number\_plate text,  violation\_id uuid,  driver\_dni text,  owner\_dni text,  PRIMARY KEY (number\_plate, violation\_id)  ) WITH CLUSTERING ORDER BY (violation\_id ASC); |
| --- |

En este caso, se usa la matrícula de los vehículos y el identificador de la infracción como clave de partición y el identificador de la infracción como clave de clustering.

## **6.14 Tabla active\_violations**

Para poder saber las infracciones que están activas en un cierto momento se crea la siguiente tabla en la que se recoge un identificador de la infracción, el estado y las fechas relevantes a dicha infracción, como es el caso de la fecha en la que se sancionó y cuando se ha resuelto. Teniendo en cuenta que son infracciones activas, no hay fecha de resolución efectiva.

| CREATE TABLE active\_violations (  violation\_id UUID,  status TEXT,  issue\_date DATE,  due\_date TIMESTAMP,  PRIMARY KEY (violation\_id)  ); |
| --- |

Para esta tabla la clave de partición es “violation\_id” ya que es lo que identifica las infracciones.

## **6.15 Tabla speed\_excess\_by\_road**

Para poder calcular el exceso de velocidad promedio en cada carretera se necesita la siguiente tabla en la que se recoge el identificador de la infracción e información sobre la carretera como el nombre de esta, la velocidad registrada y la velocidad límite que esta tiene.

| CREATE TABLE speed\_excess\_by\_road (  road\_name TEXT,  violation\_id UUID,  speed INT,  speed\_limit INT,  PRIMARY KEY (road\_name, violation\_id)  ); |
| --- |

Para hacer la partición se ha elegido la clave del nombre de la carretera y como clave de clustering se ha elegido el identificador de la infracción.

# **7. PROCESO DE MIGRACIÓN Y CARGA DE DATOS**

Inicialmente se reciben los datos en un archivo JSON de exportación el que se recogen todos los datos existentes en la base de datos de MongoDB. Tal cual como se recibe no se puede trabajar con dichos datos debido a que la librería de python utilizada no reconocía correctamente el formato del archivo, por lo que hay que adaptar su estructura para poder empezar a usarlos.

Para ello se usa un código para formatear dichos datos y darles una estructura con la que posteriormente se pueda migrar a la nueva estructura creada y adaptada a las necesidades de Apache Cassandra. Este proceso es el que se hace en el fichero de *formatted\_json.py* que se ha incluido en esta entrega.

Posteriormente, viene el proceso de adaptar ya propiamente dicho esos datos a la nueva estructura. Para ello nos basamos en un código que se incluye en la entrega llamado *migration\_mdb\_to\_cas.py* con el que se rellenaron una primera instancia de las tablas. Como se puede apreciar, en ese archivo hay tablas que no se encuentran en el diseño final y tablas que faltan, esto se debe a que tras este primer paso, se modificaron por consola aquellas tablas que se consideró que no estaban ajustadas lo suficiente a los casos de uso, haciéndose fuera de lo que sería este código.

Pese a esos cambios posteriores, ese código fue de ayuda para poder rellenar las primeras tablas y empezar a estructurar los datos dentro de nuestra base de datos nueva, pudiendo así probar las consultas y comprobar los resultados obtenidos.

Con sucesivas iteraciones se consiguieron las versiones de las tablas de las que se dispone al momento de la entrega y con las cuales se da por estructurados los datos. De esta forma, se subieron a Cassandra las últimas versiones de las tablas, probando las consultas y que los datos obtenidos eran los que tenían que ser, dando así por finalizada la migración de los datos.

En este proceso se encontraron varias problemáticas, desde problemas con la ordenación como se ha mencionado en el apartado de las consultas, columnas de datos en las que todos su valores eran nulos y casos en los que era necesario añadir columnas nuevas con cálculos precalculados para facilitar así luego la implementación de las consultas. Este último caso puede tener como ejemplo el hecho de precalcular el exceso medio de velocidad desde la velocidad de la carretera y la velocidad límite e incluirlo directamente en una columna para así tener el valor y no tener que incluirlo en posteriores consultas de datos.

Adicionalmente, en algunos de los archivos de datos ha sido necesario modificar la tipología de algunas de las columnas que había ya que esos datos no se encontraban en el formato que se necesitaba tener para poder trabajar con ellos dentro de Cassandra.

# **8. RESULTADOS OBTENIDOS**

En general se han obtenido todos los datos que se pedían con los casos de uso, pudiendo observar de primera mano que se obtenían datos con sentido y estructura como la pedida.

Sin embargo, debido a las anomalías encontradas, no se han podido ofrecer los datos finales de una manera perfecta. Por los problemas de ordenación pese a obtener los datos, estos casos no muestran los que más valor tienen, obligando así a buscar el valor, lo que ralentiza la obtención de datos de esa consulta. También, problemas asociados como el cálculo del porcentaje de infracciones que se producen por conductores que no son dueños de los vehículos, nos hemos visto limitados por la estructura de Cassandra, no pudiendo obtener directamente este valor y necesitando hacer un cálculo externo del mismo.

La pruebas que se han realizado para ver el correcto funcionamiento de todo este sistema para la base de datos de Cassandra han sido las distintas consultas realizadas y la evaluación de los resultados que se obtenían en cada iteración de mejora. En un principio estas consultas no arrojaban resultados ya que necesitaban cambios y ajustes en las tablas como se ha mencionado, posteriormente necesitaba refinar los datos en sí de las tablas para obtener los datos y tenerlos disponibles para realizar las consultas, y posteriormente el refinamiento de las consultas para que sean la solución óptima para los distintos casos de uso.

Una posible solución al fallo encontrado con la ordenación hubiese sido el incluir una cláusula WHERE en cada una de ellas para así obtener los datos ordenados. No hemos realizado eso porque hemos entendido que cambia el sentido de la consulta y lo que se pretende obtener con ella, prefiriendo mostrar el caso en el que se obtienen los datos que se pedían en el caso de uso aunque dichos datos no estén ordenados.

# **9. CONCLUSIONES**

A modo de conclusión, creemos que se ha obtenido una solución bastante óptima para los casos que se pedían, aún cuando ha habido ciertas anomalías que se han tenido que solventar por el camino. Se ha podido apreciar que hacer todo este proceso de estructuración y de migración de datos ha sido un poco más sencillo y estructurado en el caso de Cassandra frente a lo que hubo que hacer para la base de datos de MongoDB, en el que debía generarse una estructura mucho más completa y compleja para poder tener los datos. En este caso el desarrollo es bastante lineal e intuitivo, lo que facilita la creación de los distintos elementos que son necesarios. También, como en este caso el preprocesado de datos ha tenido menos carga dentro del proyecto, nos hemos podido centrar más en lo que es propiamente Cassandra y su estructura, llegando así más rápidamente a una solución óptima y un entendimiento mejor del camino a seguir.

# **10. ANEXOS**

## **11.1 Tablas iniciales de MongoDB**

| Table road {  road\_id INT [pk]  name VARCHAR(50)  speed\_limit INT  }  Table radar {  radar\_id INT [pk]  road\_id INT [ref: > road.road\_id]  mileage INT  direction VARCHAR(20)  speed\_limit INT  }  Table vehicle {  number\_plate VARCHAR(20) [pk]  make VARCHAR(50)  model VARCHAR(50)  colour VARCHAR(30)  power INT  chassis\_number VARCHAR(50)  registry\_date DATE  }  Table person {  dni VARCHAR(20) [pk]  name VARCHAR(50)  surname VARCHAR(50)  sec\_surname VARCHAR(50)  address VARCHAR(100)  town VARCHAR(50)  phone\_number VARCHAR(20)  email VARCHAR(50)  birthdate DATE  }  Table driver {  dni VARCHAR(20) [pk, ref: > person.dni]  license\_type VARCHAR(10)  license\_date DATE  }  Table owner {  dni VARCHAR(20) [pk, ref: > person.dni]  }  Table violation {  violation\_id INT [pk]  number\_plate VARCHAR(20) [ref: > vehicle.number\_plate]  radar\_id INT [ref: > radar.radar\_id]  rec\_id INT [ref: > record.rec\_id]  date DATE  time TIME  speed INT  }  Table roadworthiness {  inspection\_id INT [pk]  number\_plate VARCHAR(20) [ref: > vehicle.number\_plate]  mot\_date DATE  shortcomings TEXT  }  Table speed\_ticket {  ticket\_id INT [pk]  violation\_id INT [ref: > violation.violation\_id]  amount DECIMAL(10, 2)  currency VARCHAR(10)  issue\_date DATE  pay\_date DATE  pay\_type VARCHAR(20)  state VARCHAR(20)  debtor\_dni VARCHAR(20) [ref: > driver.dni]  }  Table record {  rec\_id INT [pk]  file VARCHAR(100)  date DATE  time TIME  speed INT  } |
| --- |