Bases de Datos

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

RTP2
Grupo 11

NoSQL
14 de diciembre de 2016

Integrante	LU	Correo electrónico
Julián Bayardo	850/13	julian@bayardo.com.ar
Christian Cuneo	755/13	chriscuneo93@gmail.com
Mauro Cherubini	835/13	cheru.mf@gmail.com
Martin Baigorria	575/14	martinbaigorria@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Índice

1. Introducción

A partir del diseño realizado anteriormente, el gobierno ahora nos ha solicitado mejorar el tiempo de ejecución de algunas consultas con el objetivo de que los oficiales de policía puedan identificar personas sospechosas rápidamente (sí, cuando se trata de combatir el delito los nanosegundos son importantes). Para ello, nuestro equipo ha decidido utilizar una base de datos no relacional, basada en documentos. Partimos del modelo conceptual desarrollado en el trabajo anterior. Para realizar el nuevo Diagrama de Interrelación de Documentos (DID) necesario para el diseño de nuestra nueva base de datos, tuvimos en cuenta las siguientes consultas que el gobierno desea realizar frecuentemente:

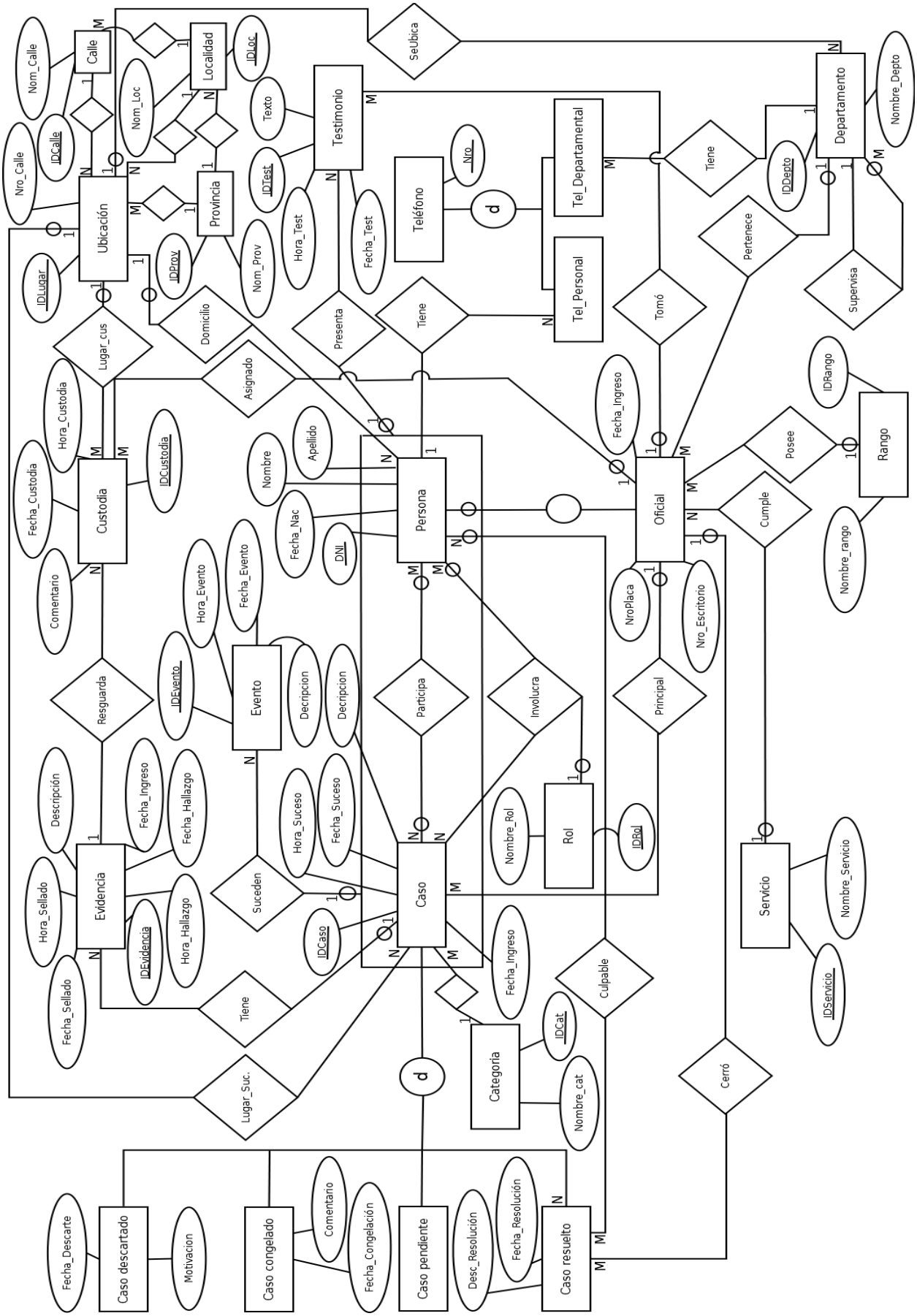
1. Cual es el numero promedio de crímenes cometidos por personas que ya han sido encontradas culpables de algún crimen?
2. Cuales son las personas que se han visto involucradas como testigos en el mayor numero de casos?
3. Cuales son los casos en los que se han visto involucradas (en cualquier rol) el mayor numero de personas?
4. Que cantidad, de cada tipo de crimen, ocurrieron en los últimos 45 días?
5. Cual es el mayor numero de crímenes cometido por alguna persona?
6. Cuales son las 10 ciudades con mayor numero de crímenes?
7. Cuales son los casos con mayor numero de evidencias?

La implementación sera realizada en MongoDB.

1.1. Supuestos

- La cantidad de casos que hay en una misma fecha es manejable, lo mismo ocurre para los casos que ocurren en una misma localidad.
- Cada categoría tiene un nombre de categoría distinto, es decir no hay dos categorías con diferente ID pero que compartan el mismo nombre.
- Cada rol tiene un nombre de rol distinto, es decir no hay dos roles con diferente ID pero que compartan el mismo nombre.
- La cantidad de roles y categorías es bastante acotada, de manera que son cantidades bastante manejables.
- A la hora de realizar aquellas consultas de las cuales se requieran los datos de ciertas personas, no es de interés devolver los datos propios de un oficial (para aquellos que lo sean), a lo sumo interesará saber si es o no es un oficial.
- El número de casos en los cuales una persona fue culpable, no es excesivamente grande.

1.2. Diagrama de Entidad Relación (DER)



1.3. Motivación del diseño

- **Caso:** Este documento esta inspirado en la entidad Caso del DER, pero con algunos agregados que si bien podrían llegar a generar redundancia sirven para optimizar las consultas solicitadas en el enunciado. Para obtener la lista de involucrados en un determinado caso, le agregamos a Caso un arreglo de DNIs que corresponden a aquellas personas que se vieron involucradas bajo cualquier rol con dicho caso, de esta forma no hace falta recorrer la tabla Involucra que teníamos a partir del DER, para saber la identidad cada involucrado. Para poder resolver cuanta evidencia hay por cada caso basta localizar el caso del cual queremos su evidencia, pues esta está embebida en el mismo por medio de un arreglo de documentos que almacena la información pertinente a cada evidencia con la que se relaciona dicho caso. Vale aclarar que si bien se podría haber modelado como anteriormente con las personas (dando un arreglo de IDs) ya que el enunciado solo nos pide saber solo la cantidad de evidencia por cada caso, debido a la intima relación de cada evidencia con su caso nos pareció mas sensato resguardarla toda allí mismo. Análogamente ocurre lo mismo con Categoría y Ubicación, ambas fueron embebidas en Caso, si bien habrá muchos casos que tengan la misma categoría o puedan llegar a coincidir en la ubicación, la redundancia se justifica en que estos son datos básicos de cada caso de manera que es normal pensar que se necesitará acceder a los mismo de manera inmediata.

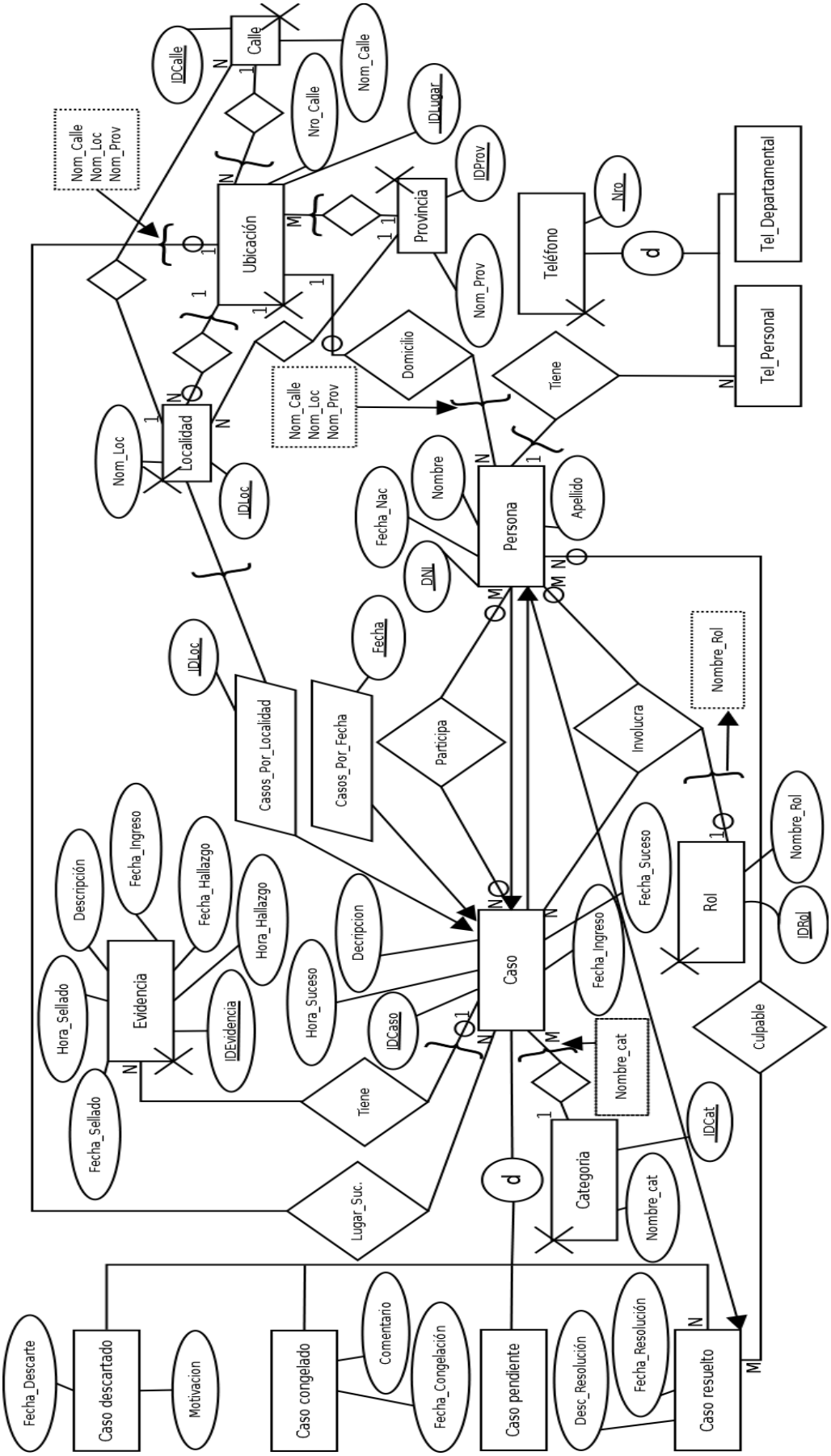
- **Persona:** Este documento esta inspirado en la entidad Persona del DER, a la cual agregamos contenido extra con el fin de optimizar algunas consultas. Para saber lo mas rápido posible cual es el número de crímenes cometidos por persona (que ya hayan cometido algún crimen) y cuales son las personas que han sido testigos en el mayor número de casos, añadimos dos arreglos *Culpable* e *Involucrado*. El arreglo *Culpable* contiene los IDs de todos los casos en los que la persona en cuestión haya sido encontrada culpable; mientras que *Involucrado* contiene para cada entrada un documento que guarda un nombre de rol y un arreglo de IDs correspondientes a los casos en los que la persona en cuestión haya estado involucrado con el rol al cual acompaña. Por ultimo nos pareció sensato embeber la ubicación y teléfono pues son datos básicos que posiblemente se quisieran acceder dado que ya se accedió al documento de una persona.

- **Casos_Por_Localidad:** La idea detrás de este documento es poder identificar rápido cuales y que cantidad de casos ocurrieron en una localidad especifica. Para ello embebemos la Localidad, teniendo así el nombre de la localidad y su ID, que a su vez será el ID de este nuevo documento. Almacenamos en él además, un arreglo con los IDs de todos los casos cuya ubicación corresponda a la localidad que posee el mismo ID que este documento; de esta manera podemos satisfacer mucho mas rápido aquellas consultas para las cuales necesitamos saber cuantos casos hay por cada localidad.

- **Casos_Por_Fecha:** Al igual que con la localidad, también se necesita obtener rápidamente los casos que ocurrieron en cierto rango de fechas. Para ello es que este documento adopta como primary key la fecha, y guarda un arreglo de documentos, cuyas entradas contienen el nombre de una categoría y un arreglo con los IDs de todos los casos correspondientes a la misma, que vale aclarar, ocurrieron en la fecha indicada por la primary key. De esta forma para saber cuantos crímenes ocurrieron entre ciertas fechas, alcanza con localizar por fecha aquellos *Casos_Por_Fecha* que correspondan y contar la cantidad de IDs de casos contenidos (incluso con la posibilidad de discriminar en la cuenta por categoría).

1.4. Diagrama de Interrelación de Documentos (DID)

El siguiente DID es un intento por ilustrar gráficamente la inspiración detrás de cada documento JSON modelado para este trabajo práctico. Vale aclarar que si bien para esto nos basamos en el apunte de DID y en las presentaciones vistas en clase, debido a la falta de ejemplos mas complejos es posible que este diagrama no respete o siga alguna convención con la cual no estamos completamente familiarizados.


Calle
IDCalle
Nom_Calle
Nro_Calle
Ubicación
Nom_Calle
Nom_Loc
Nom_Prov
IDLugar
Provincia
IDProv
Nom_Prov
Teléfono
Nro
Tel_Departamental
Tel_Personal
Domicilio
Tiene
Localidad
Nom_Loc
IDLoc
Nom_Calle
Nom_Loc
Nom_Prov
Persona
Fecha_Nac
Nombre
Apellido
DNI
Casos_Por_Localidad
Casos_Por_Fecha
Fecha
Participa
Involucra
Nombre_Rol
Rol
IDRol
Culpable
Evidencia
Descripción
Fecha_Ingreso
Fecha_Hallazgo
Hora_Hallazgo
Hora_Sellado
Fecha_Sellado
IDEvidencia
Hora_Suceso
Decripcion
IDCaso
Caso
Fecha_Ingreso
Fecha_Suceso
Tiene
Lugar_Suc.
Categoria
Nombre_cat
IDCat
Nombre_cat
Caso descartado
Fecha_Descarte
Motivacion
Caso congelado
Comentario
Fecha_Congelación
Caso pendiente
Caso resuelto
Desc_Resolución
Fecha_Resolución

2. Conclusiones

En este segundo trabajo práctico, en base al modelo conceptual anteriormente desarollado y a los nuevos requerimentos del cliente, pasamos a una base de datos NoSQL orientada a documentos. La idea detrás de este trabajo fue la de poder optimizar determinadas consultas que por diversas razones necesitamos potencialmente acelerar en relación al tiempo de consulta requerido en un modelo relacional, evitando los JOINs. Pudimos notar durante el desarrollo de este trabajo que para poder realizar dicha tarea debíamos modelar los documentos de manera inteligente, para lo que debíamos desestimar la importancia de la redundancia y espacio en disco, que adquirían otra importancia en el trabajo anterior.

Ciertas consultas como las de "todos los documentos que son un máximo global según algún criterio", son más complicadas de expresar en este paradigma: nos requieren utilizar colecciones intermedias para poder guardar los resultados del mapReduce. Si bien esto es algo que implícitamente estaría sucediendo en el paradigma relacional cuando utilizamos una subquery (que hubiera sido la forma de resolver el problema), la sintaxis de la expresión y la carga (en términos de complejidad expuesta al usuario) de la misma resultan ampliamente mayores en MongoDB: nos vemos forzados a crear una colección intermedia, y borrarla luego. A cambio de esta inconveniencia, ganamos en escalabilidad: las consultas mapReduce que escribimos son masivamente paralelizables de una forma casi gratuita en términos de sintaxis, que no es lo que ocurre con SQL.

En conclusión, vimos que MongoDB simplifica ampliamente el proceso de escritura de sistemas escalables: pudimos armar una base de datos optimizada para el caso de uso particular de una forma muy simple, y escribir consultas sobre la misma de una forma más simple aun. Lo interesante es que el esfuerzo de manejar los mecanismos de control de concurrencia y todos sus problemas asociados (así como la transferencia de información entre servidores, etcétera) fue completamente eliminado, permitiéndonos adoptar estrategias de computo embarrassingly parallel de una forma muy sencilla. Además, la flexibilidad del modelo map reduce nos permite expresar cómputos muy complejos de formas comparativamente simples contrastando con la alternativa de utilizar SQL.

3. JSON Schemas

3.1. Persona

```
{
  "type": "object",
  "properties": {
    "dni": { "type": "number" },
    "nombre": { "type": "string" },
    "apellido": { "type": "string" },
    "fecha_nacimiento": { "type": "date" },
    "involucrado": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "rol": { "type": "string" },
          "caso": { "type": "number" }
        }
      }
    },
    "culpable": {
      "type": "array",
      "items": { "type": "number" }
    },
    "telefono": { "type": "number" },
    "domicilio": {
      "type": "object",
      "properties": {
        "calle": { "type": "string" },
        "numero": { "type": "number" },
        "localidad": { "type": "string" },
        "provincia": { "type": "string" }
      }
    },
    "oficial": { "type": "boolean" }
  },
  "required": ["dni", "nombre", "apellido"]
}
```

3.2. Caso

```
{
  "type": "object",
  "properties": {
    "id": { "type": "number"},
    "timestamp_suceso": { "type": "timestamp" },
    "descripcion": { "type": "string" },
    "personas": {
      "type": "array",
      "items": { "type": "number" }
    },
    "estado": {
      "type": "string",
      "enum": [
        "pendiente",
        "congelado",
        "descartado",
        "resuelto"
      ]
    },
    "categoria": { "type": "string" },
    "lugar_suceso": {
      "type": "object",
      "properties": {
        "calle": { "type": "string" },
        "numero": { "type": "number" },
        "localidad": { "type": "string" },
        "provincia": { "type": "string" }
      },
      "required": ["calle", "numero", "localidad", "provincia"]
    },
    "evidencia": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "descripcion": { "type": "string" },
          "timestamp_ingreso": { "type": "timestamp" },
          "timestamp_hallazgo": { "type": "timestamp" },
          "timestamp_sellado": { "type": "timestamp" }
        },
        "required": ["descripcion", "timestamp_ingreso", "timestamp_hallazgo"]
      }
    }
  },
  "required": ["id", "timestamp_suceso", "estado", "categoria", "lugar_suceso"]
}
```

3.3. Casos por localidad

```
{
  "type": "object",
  "properties": {
    "localidad": { "type": "string" },
    "casos": {
      "type": "array",
      "items": { "type": "number" }
    },
    "required": ["localidad", "casos"]
  }
}
```

3.4. Casos por fecha

```
{
  "type": "object",
  "properties": {
    "fecha": { "type": "date" },
    "crimenes": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "categoria": { "type": "string" },
          "casos": {
            "type": "array",
            "items": { "type": "number" }
          }
        }
      }
    },
    "required": ["fecha", "crimenes"]
  }
}
```

4. Map Reduce

```
// Numero promedio de crimenes cometidos por personas que ya han sido encontradas culp
var map1 = function() {
  if (this.culpable && this.culpable.length > 0) {
    emit(1, { qty: 1, sum: this.culpable.length })
  }
}

var reduce1 = function(key, values) {
  reducedVal = { sum: 0, qty: 0 };

  for (var idx = 0; idx < values.length; idx++) {
    reducedVal.sum += values[idx].sum;
    reducedVal.qty += values[idx].qty;
  }
  return reducedVal;
}

var finalize1 = function(key, reduced) {
  return reduced.sum / reduced.qty;
}

db.personas.mapReduce(
  map1,
  reduce1,
  {
    out: "promedio_crimenes_cometidos",
    finalize: finalize1
  }
);

// Personas involucradas como testigos en la mayor cantidad de casos.
var map2 = function() {
  var ncasos = 0;
  if (this.involucrado) {
    for (var idx = 0; idx < this.involucrado.length; idx++) {
        if (this.involucrado[idx].rol == 'testigo') {
        ncasos++;
      }
    }

  }
  if (ncasos > 0) { emit(ncasos, { personas : [this.dni] }); }
}

var reduce2 = function(key, values) {
  var res = { personas: [] }
  values.forEach(function(e){ res.personas.push.apply(res.personas,e.personas) })
  return res;
}

db.personas.mapReduce(map2, reduce2, { out: 'aux_2' });
db.createCollection('personas_involucradas_como_testigos');
db.personas_involucradas_como_testigos.insertMany(
  // Ordeno los resultados de mayor a menor, me quedo con el primero y me fijo las per
```

```
  // Inserto cada persona como un objeto { dni: numero } a la coleccion final
  db.aux_2.find({}).sort({_id:-1})[0].value.personas.map(function(e){ return {dni: e}
)
db.aux_2.drop() // Elimino coleccion auxiliar


//Casos en los que se han visto involucradas el mayor numero de personas.
var map3 = function() {
  emit(this.personas.length, { casos: [this.id] });
}

var reduce3 = function(key, values) {
  var res = { casos: [] }
  values.forEach(function(e){ res.casos.push.apply(res.casos,e.casos) })
  return res;
}

db.casos.mapReduce(map3, reduce3, { out: 'aux_3' });
db.createCollection('casos_con_mayor_numero_de_personas_involucradas');
db.casos_con_mayor_numero_de_personas_involucradas.insertMany(
  // Ordeno los resultados de mayor a menor, me quedo con el primero y me fijo los cas
  // Inserto cada caso como un objeto { caso: numero } a la coleccion final
  db.aux_3.find({}).sort({_id:-1})[0].value.casos.map(function(e){ return {caso: e}})
)

db.aux_3.drop() // Elimino coleccion auxiliar

// Cantidad de crimenes por localidad y por ano.
var map4 = function() {
  year = this.timestamp_suceso.getFullYear();
  d = {}
  d[year] = 1;
  emit(this.lugar_suceso.localidad, d);
}

var reduce4 = function(key, values) {
  d = {}
  for (i in values) {
    for (year in values[i]) {
      if (year in d) {
        d[year] = d[year]+1;
      } else {
        d[year] = 1;
      }
    }
  }
 return d;
}

db.casos.mapReduce(map4, reduce4, { out: 'crimenes_por_localidad_y_ano' });

// Mayor numero de crimenes cometido por alguna persona.
var map5 = function() {
  if (this.culpable) {
    emit(1, this.culpable.length);
  }
```

```
}

var reduce5 = function(key, values) {
  return values.reduce(function(pv, cv) { return (pv > cv) ? pv : cv ; }, 0);
}

db.personas.mapReduce(map5, reduce5, { out: 'mayor_numero_de_crimenes_cometidos_por_un

// Cantidad total de evidencias por caso.
var map6 = function() {
  if (this.evidencia) {
    emit(this.id, this.evidencia.length);
  }
}

var reduce6 = function(key, values) {
  return values.reduce(function(pv, cv) { return pv + cv; }, 0);
}

db.casos.mapReduce(map6, reduce6, { out: 'cantidad_total_de_evidencias_por_caso' });

// Las 10 ciudades con mayor numero de crimenes.
var map7 = function() {
  emit(this.lugar_suceso.localidad, 1)
}

var reduce7 = function(key, values) {
  return values.reduce(function(pv, cv) { return pv + cv; }, 0);
}

db.casos.mapReduce(map7, reduce7, { out: 'ciudades_con_mayor_numero_de_crimenes' })

db.promedio_crimenes_cometidos.find()
db.personas_involucradas_como_testigos.find().sort( { value: -1 } );
db.casos_con_mayor_numero_de_personas_involucradas.find().sort( { value: -1 } );
db.crimenes_por_localidad_y_ano.find()
db.mayor_numero_de_crimenes_cometidos_por_una_persona.find()
db.cantidad_total_de_evidencias_por_caso.find()
db.ciudades_con_mayor_numero_de_crimenes.find().sort( { value: -1 } );
```