Gerard Simon Estadella 1671196

Max González Andrés 1671640

1. Disseny de l’esquema de la base de dades

- **Analitzar l’estructura de les dades i determinar el tipus de relació entre restaurants i inspeccions (One-to-Few, One-to-Many, One-to-Millions).** La relació entre restaurants i inspeccions serà one-to-many, ja que un restaurant pot tenir múltiples inspeccions (en concret té una mitjana de 2,5 inspeccions per restaurant) i una inspecció només pot estar associada a un restaurant.
- **Justificar l’elecció de referències (restaurant_id) en lloc de documents embeguts. També es pot decidir crear una nova col·lecció que no utilitzi referències i incorpori els documents embeguts. Justificar la decisió.** L’ús de referències és la millor opció, perquè si féssim servir documents embeguts, a mesura que es fessin moltes inspeccions d’un mateix restaurant el document es faria enorme, i a més, cada vegada que volguéssim veure una inspecció concreta hauríem de carregar totes les altres també.

Definir un esquema de validació per a ambdues col·leccions.

restaurants:

```
{
  "$jsonSchema": {
    "bsonType": "object",
    "required": ["name", "address", "type_of_food", "rating"],
    "properties": {
      "name": { "bsonType": "string" },
      "address": { "bsonType": "string" },
      "type_of_food": { "bsonType": "string" },
      "rating": {
        "bsonType": "int",
        "minimum": 0,
        "maximum": 5
      },
      "outcode": { "bsonType": "string" },
      "postcode": { "bsonType": "string" }
    }
  }
}
```

Es considera que els camps name, address, type_of_food i rating són obligatoris per descriure correctament un restaurant. També es limita el valor de rating a un enter entre 0 i 5 per mantenir valors consistents.

```
{
  "$jsonSchema": {
    "bsonType": "object",
    "required": ["restaurant_id", "date", "result"],
    "properties": {
      "restaurant_id": { "bsonType": "string" },
      "date": { "bsonType": "string" },
      "result": { "bsonType": "string" },
      "certificate_number": { "bsonType": "string" },
      "business_name": { "bsonType": "string" },
      "sector": { "bsonType": "string" },
      "address": { "bsonType": "object" }
    }
  }
}
```

Els camps restaurant_id, date i result són obligatoris per reflectir correctament cada inspecció. També es valida el tipus de dades de la resta de camps per evitar inconsistències.

2. Implementació de consultes en MongoDB

- **Buscar tots els restaurants d'un tipus de menjar específic (ex. "Chinese").**
  Dins la col·lecció de restaurants, filtrarem pel camp de type_of_food que el seu valor sigui Chinese.

```
db.restaurants.find({
    "type_of_food": "Chinese"
})
```

  Sortida:

```
db.restaurants.find({
    "type_of_food": "Chinese"
})

{
  _id: ObjectId('55f14312c7447c3da7051b26'),
  URL: 'http://www.just-eat.co.uk/restaurants-cn-chinese-cardiff/menu',
  address: '228 City Road',
  'address line 2': 'Cardiff',
  name: '.CN Chinese',
  outcode: 'CF24',
  postcode: '3JH',
  rating: 5,
  type_of_food: 'Chinese'
}
{
  _id: ObjectId('55f14312c7447c3da7051b2f'),
  URL: 'http://www.just-eat.co.uk/restaurants-1-2-3-chinese-rowlands-gill/menu',
  address: 'Unit 4 Spencer House',
  'address line 2': 'Swalwell',
  name: '1 2 3 Chinese',
  outcode: 'NE16',
  postcode: '3DS',
  rating: 4.5,
  type_of_food: 'Chinese'
}
```

  ...

- **Llistar les inspeccions amb violacions, ordenades per data.**

Dins la col·lecció inspeccions, filtrarem pel camp result, on el valor sigui diferent de No Violation Issued i amb el mètode sort els ordenarem en ordre ascendent

```
db.inspections.find({ "result": { $ne: "No Violation Issued" } }).sort({ "date": 1 })
```

Sortida:

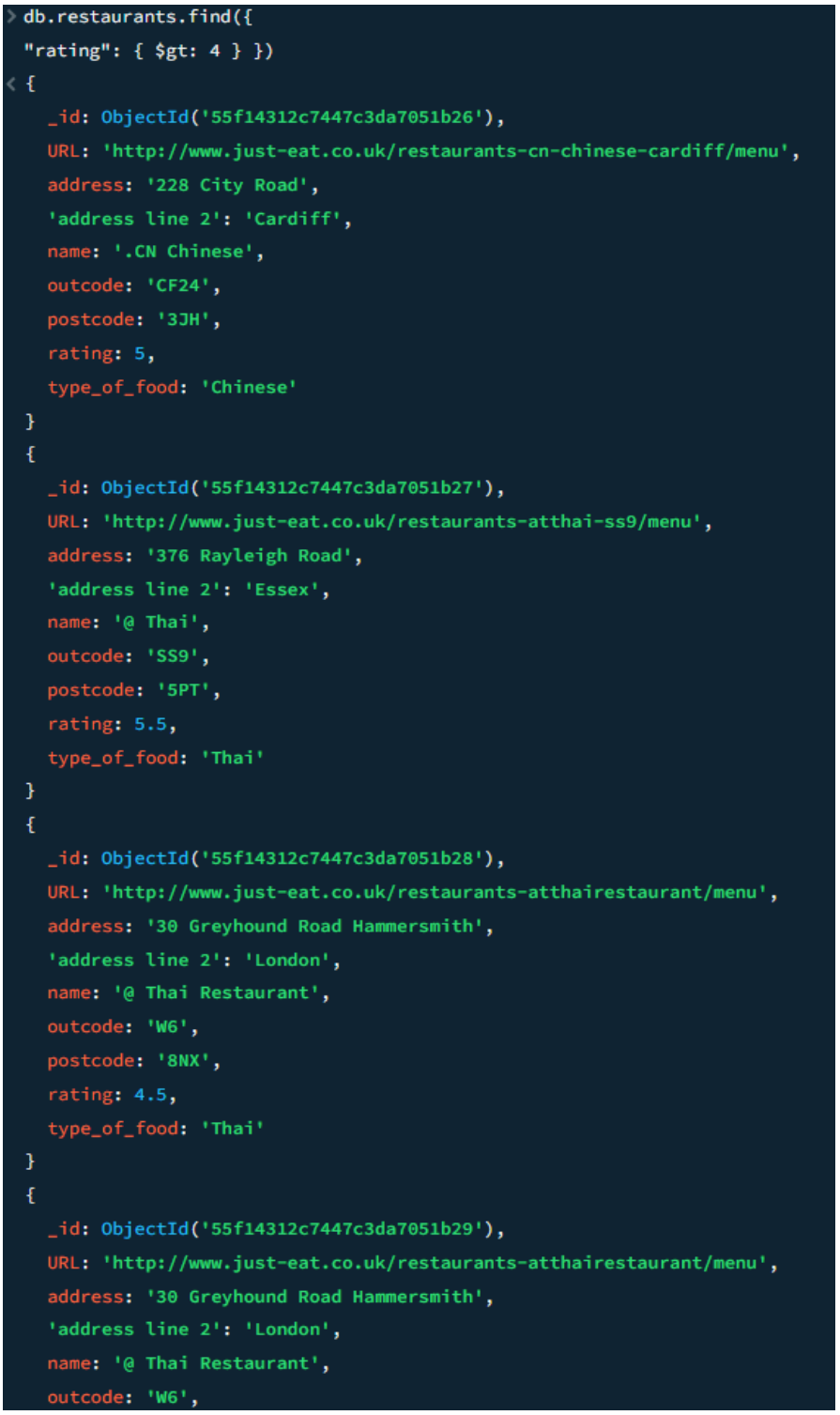
```
> db.inspections.find({ "result": { $ne: "No Violation Issued" } }).sort({ "date": 1 })

< {
    _id: ObjectId('56d61034a378eccde8a8c5ad'),
    id: '50402-2015-ENFO',
    certificate_number: 9276754,
    business_name: 'BHOJAN INDIAN TAKEAWAY',
    date: 'Apr 01 2022',
    result: 'Fail',
    sector: 'Cigarette Retail Dealer - 127',
    address: {
      city: 'DARTFORD',
      zip: '1NP',
      street: 'HAWLEY ROAD',
      number: '3'
    },
    restaurant_id: '55f14313c7447c3da7052445'
  }
  {
    _id: ObjectId('56d61034a378eccde8a90a1b'),
    id: '55006-2015-ENFO',
    certificate_number: 9309882,
    business_name: 'BIG BOY PIZZA',
    date: 'Apr 01 2022',
    result: 'Fail',
    sector: 'Grocery-Retail - 808',
    address: {
      city: 'LONDON',
      zip: '4JP',
      street: 'BATTERSEA PARK ROAD',
      number: '52'
    },
    restaurant_id: '55f14313c7447c3da705246a'
  }
  {
    _id: ObjectId('56d61034a378eccde8a91cbe'),
    id: '42633-2015-ENFO',
    certificate_number: 9273739,
    business_name: 'A WORLD OF FL@VOURS',
    date: 'Apr 01 2022',
    result: 'Warning Issued',
```

- **Trobar restaurants amb una qualificació superior a 4.**
  Dins la col·lecció restaurants filtrarem pel camp rating, on el valor sigui més gran que 4.

```
db.restaurants.find({
"rating": { $gt: 4 } })
```

  Sortida:



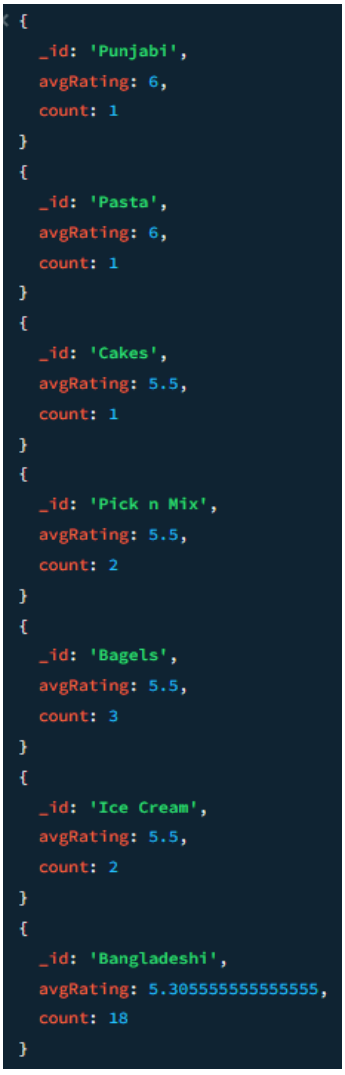
```
> db.restaurants.find({
  "rating": { $gt: 4 } })
< {
    _id: ObjectId('55f14312c7447c3da7051b26'),
    URL: 'http://www.just-eat.co.uk/restaurants-cn-chinese-cardiff/menu',
    address: '228 City Road',
    'address line 2': 'Cardiff',
    name: '.CN Chinese',
    outcode: 'CF24',
    postcode: '3JH',
    rating: 5,
    type_of_food: 'Chinese'
  }
  {
    _id: ObjectId('55f14312c7447c3da7051b27'),
    URL: 'http://www.just-eat.co.uk/restaurants-atthai-ss9/menu',
    address: '376 Rayleigh Road',
    'address line 2': 'Essex',
    name: '@ Thai',
    outcode: 'SS9',
    postcode: '5PT',
    rating: 5.5,
    type_of_food: 'Thai'
  }
  {
    _id: ObjectId('55f14312c7447c3da7051b28'),
    URL: 'http://www.just-eat.co.uk/restaurants-atthairestaurant/menu',
    address: '30 Greyhound Road Hammersmith',
    'address line 2': 'London',
    name: '@ Thai Restaurant',
    outcode: 'W6',
    postcode: '8NX',
    rating: 4.5,
    type_of_food: 'Thai'
  }
  {
    _id: ObjectId('55f14312c7447c3da7051b29'),
    URL: 'http://www.just-eat.co.uk/restaurants-atthairestaurant/menu',
    address: '30 Greyhound Road Hammersmith',
    'address line 2': 'London',
    name: '@ Thai Restaurant',
    outcode: 'W6',
```

3. ÚS d'agregacions

- **Agrupar restaurants per tipus de menjar i calcular-ne la qualificació mitjana.**
  Dins la col·lecció restaurants agrupem pel camp type_of_food, calculem la mitjana del camp rating i comptem el nombre de restaurants de cada tipus, després els ordenem en ordre descendent.

```
db.restaurants.aggregate([
  {
    $group: {
      _id: "$type_of_food",
      avgRating: { $avg: "$rating" },
      count: { $sum: 1 }
    }
  },
  {
    $sort: { avgRating: -1 }   }
])
```

  Sortida:



  ...

- **Contar el número de inspecciones por resultado y mostrar los porcentajes.**

```
db.inspections.aggregate([
  {
    $group: {
      _id: "$result",
      count: { $sum: 1 }
    }
  },
  {
    $group: {
      _id: null,
      total: { $sum: "$count" },
      results: { $push: { result: "$_id", count: "$count" } }
    }
  },
  { $unwind: "$results" },
  {
    $project: {
      _id: 0,
      result: "$results.result",
      count: "$results.count",
      percentage: {
        $multiply: [
          { $divide: ["$results.count", "$total"] },
          100
        ]
      }
    }
  }
])
```

Utilitzem una agregació per comptar el nombre d'inspeccions segons el resultat i calcular-ne el percentatge. Primer, amb **$group**, s'agrupen totes les inspeccions pel camp **result** i es compta quantes inspeccions té cada resultat amb **$sum**. A continuació, es torna a utilitzar **$group** per sumar el total d'inspeccions i guardar els resultats en un array anomenat **results**. Després, amb **$unwind**, es descompon l'array **results** per poder treballar amb cada resultat individualment. Finalment, amb **$project**, es mostra el nom del resultat, el nombre d'inspeccions i el percentatge que representa, calculat amb **$divide** i **$multiply** per obtenir el percentatge respecte al total.

Sortida:

```
{
  result: 'Warning Issued',
  count: 1280,
  percentage: 20.09419152276295
}
{
  result: 'Pass',
  count: 1259,
  percentage: 19.76452119309262
}
{
  result: 'Violation Issued',
  count: 1291,
  percentage: 20.266875981161693
}
{
  result: 'Fail',
  count: 1280,
  percentage: 20.09419152276295
}
```

- **Unir restaurantes con sus inspecciones utilizando `$lookup`.**

```
db.restaurants.aggregate([
  {
    $addFields: {
      id_string: { $toString: "$_id" }
    }
  },
  {
    $lookup: {
      from: "inspections",
      localField: "id_string",
      foreignField: "restaurant_id",
      as: "inspections"
    }
  },
  {
    $project: {
      id_string: 0
    }
  },
  { $limit: 5 }
])
```

Aquesta consulta utilitza una agregació per unir els restaurants amb les seves inspeccions.

Primer, amb **$addFields**, s'afegeix un camp nou anomenat **id_string**, que conté l'**_id** del restaurant convertit a format string. Això es fa perquè el camp **restaurant_id** de la col·lecció **inspections** és de tipus string i cal que tinguin el mateix format per comparar-los.

A continuació, amb l'etapa **$lookup**, es busca dins la col·lecció **inspections** totes les inspeccions que tenen un **restaurant_id** igual al **id_string** del restaurant, afegint-les en un nou camp anomenat **inspections**.

Després, amb **$project**, s'elimina el camp auxiliar **id_string** per no mostrar-lo en el resultat final. Finalment amb **$limit**, es limiten els resultats a només 5 restaurants amb les seves inspeccions associades.

Sortida:

… (fins a 5 restaurants)

4. Optimització de Rendiment

Per a optimitzar el rendiment de les consultes, hem decidit crear els següents índex:

Sobre restaurants:

- Índex sobre type_of_food:
  Gràcies a aquest índex aconseguim optimitzar la consulta de cerca de restaurants xinesos. Després de crear l'índex, MongoDB passa d'examinar tots els 2548 documents/restaruants (COLLSCAN) a examinar només els 174 restaurants rellevants mitjançant un IXSCAN. Això redueix el temps d'execució de 2 ms a 1 ms, millorant l'eficiència i preparant la consulta per escalar millor amb col·leccions més grans.

- Índex sobre rating:
  L'hem creat per a optimitzar la consulta sobre els restaurants amb una qualificació superior de 4. Després de crear-lo, passem d'examinar tots els 2548 restaurants (COLLSCAN) a examinar només els 2228 restaurants rellevants mitjançant un IXSCAN. Ja que la reducció de documents examinats no és gaire notòria, el temps d'execució és similar.

Sobre inspeccions:

- Índex compost sobre result + date:
  L'hem decidit crear per a optimitzar la consulta que llista les inspeccions que han violat alguna norma ordenades per data ascendent. Després de crear l'índex compost sobre els camps result i date, MongoDB passa d'examinar tots els 6370 documents d'inspeccions (COLLSCAN) a examinar només els 5110 documents amb IXSCAN. Tot i que en aquest cas ens hem trobat que el temps després de crear l'index s'ha incrementat en comptes de reduir-se. Creiem que és perquè, en utilitzar aquest índex compost, MongoDB necessita fer passos extra com el FETCH (recuperar els documents complets) i el SORT (ordenar per data). En col·leccions petites, aquest sobrecost pot ser superior al d'un COLLSCAN directe.

- Índex sobre restaurant_id
  L'hem creat per a optimitzar el lookup entre inspeccions i restaurants. Abans de crear-lo, s'havia d'examinar tota la col·lecció inspections fent un collection scan complet (COLLSCAN), havent d'examinar 31.850 documents en total.
  Després de crear l'índex sobre restaurant_id, hem reduit dràsticament el nombre de documents examinats a només 15 documents (els 5 restaurants retornats (limit especificat) × coincidències directes per índex), sense cap COLLSCAN.
  El temps d'execució es redueix, passant d'uns 21 ms a 2 ms.

5. Estrategies d’escalabilitat

Sharding:
Proposem aplicar sharding sobre la col·lecció inspections utilitzant el camp restaurant_id xifrat amb hash per així repartir els documents de manera uniforme entre els shards, millorant el rendiment de consultes per restaurant.

Replicació:
Configurariem un Replica Set amb 1 node primari i 2 secundaris per garantir l’estabilitat i l’alta disponibilitat de la base de dades. Si cau el primary, un dels secundaris prendrà automàticament el relleu, assegurant la continuïtat del servei.

Possibles colls d’ampolla i solucions:

1. Consultes lentes per COLLSCAN:
La solució és crear índexs adequats, com ja s’ha fet amb els camps type_of_food, rating i result.

2. Ús intensiu de $lookup:
El $lookup pot ser lent si ha de buscar en una col·lecció gran sense índex, ja que ha d’escanejar tota la col·lecció per trobar coincidències. Per evitar això hem creat un índex sobre el camp restaurant_id a inspections.

3. Gran volum d’escriptures:
El sharding permet distribuir les operacions d’escriptura entre diversos nodes, evitant que un sol node es saturi.

4. Caiguda de nodes:
El Replica Set amb almenys 3 nodes garanteix un recanvi automàtic de node i alta disponibilitat.

5. Ordenacions lentes amb sort:
Hem aplicat índexs compostos que inclouen els camps utilitzats en les ordenacions, millorant així el rendiment.