

BASE DE DATOS 2



Agencia de Turismo

INTRODUCCIÓN

En el siguiente proyecto vamos a tratar una problemática (aclarada más adelante), mediante una base de datos no SQL, usando MongoDB y Qgis (debido a que nuestro proyecto incluye el uso de datos geoespaciales).

Ahora vamos a mencionar algunas características a tener en cuenta a la hora de usar una DB no-relacional:

1-Productividad del desarrollo: interacción simple entre la aplicación y la base de datos

2-Datos a gran escala: Dar posibilidad a la carga masiva de datos en un sistema distribuido (clúster).

PROBLEMÁTICA

La problemática que nosotros vamos a tratar en este proyecto, es sobre una base de datos de una agencia de turismo, llamada Lucrisca.

Antes de iniciar el proyecto realizaremos una breve explicación de cómo está compuesta la agencia de turismo:

La agencia está dividida en diferentes bloques. Por un lado, tenemos los empleados de la agencia, que pueden ser de 4 tipos: Guía, Conductor, Administrador y Seguridad. Asimismo, debemos tener en cuenta que, dentro de la categoría **CONDUCTOR**, se les asignará un vehículo; los cuales son de tipo: "Colectivo", "Combi", "Trafic".

Cada **VEHÍCULO** tendrá distintas especificaciones como: Capacidad de combustible y capacidad de personas entre otras.

Además, cabe mencionar que cada **CONDUCTOR**, va a poseer una licencia obligatoria que lo habilita para el manejo de estos **VEHÍCULOS**. Y a su vez a cada vehículo se le va a asignar una ruta, la cual puede variar. Disponemos de 4 rutas, cada una en formato GeoJSON y con su respectivo mapeo en QGIS (Imágenes vectoriales)

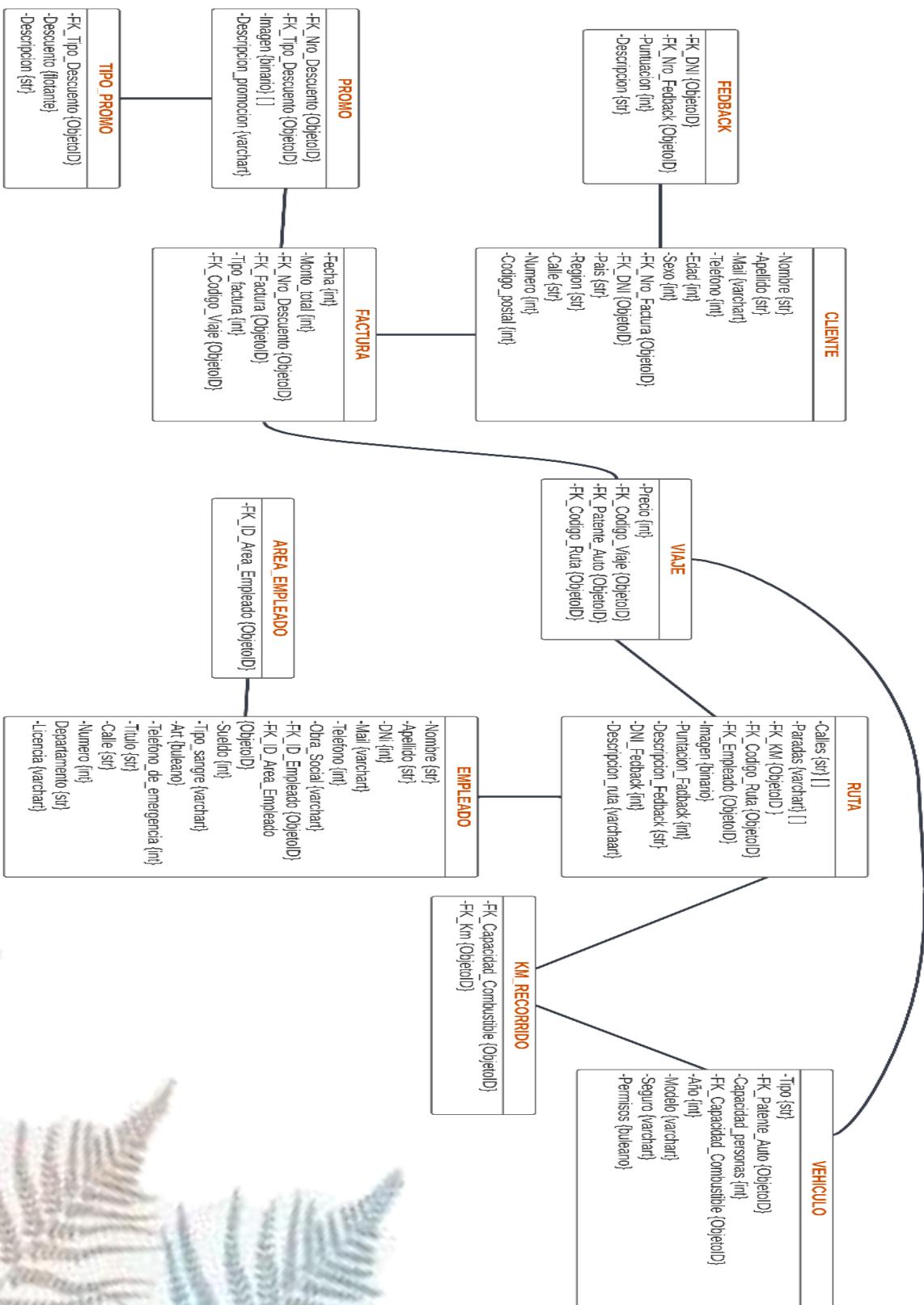
Otro bloque son los clientes los cuales deben estar registrados en el sitio de la agencia (sus datos son almacenados en nuestra base de datos). En el caso de los clientes nuevos, son dados de alta y cargados en el sistema con sus respectivos datos.

La agencia cuenta con diferentes promociones para los clientes, con sus respectivos descuentos. El cliente (el cual ya está registrado) va a poder acceder a elegir el viaje, promociones, y realizar el pago. Una vez que se efectúe el pago del viaje, se emite la factura vía mail. Por último, el cliente tiene la opción (después de realizar el viaje) de mandar un feedback al sistema.

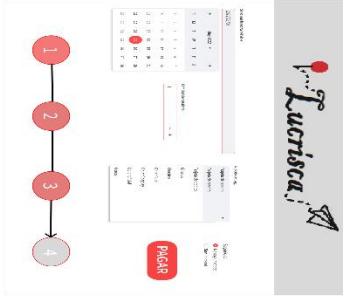
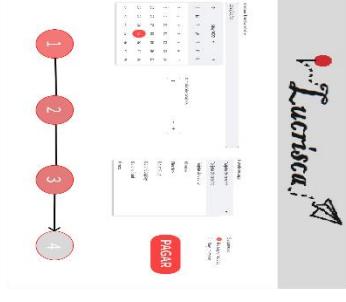
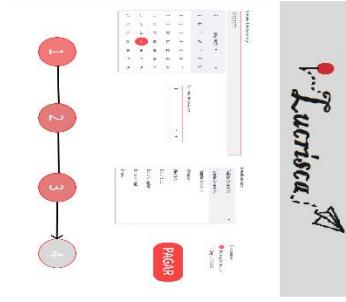
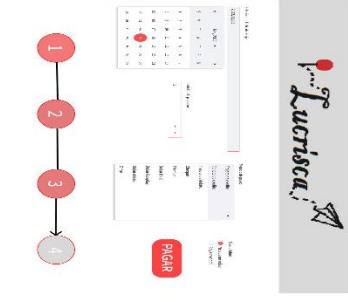
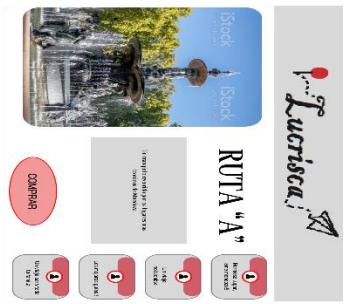
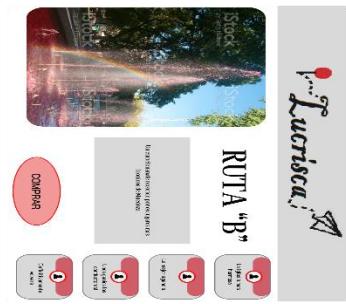
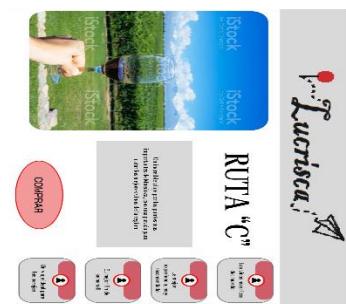
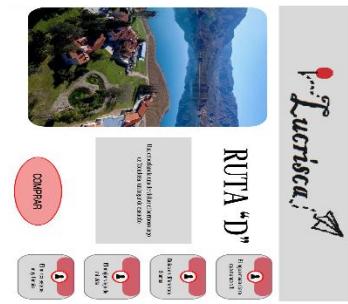
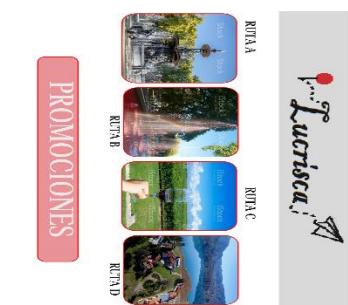
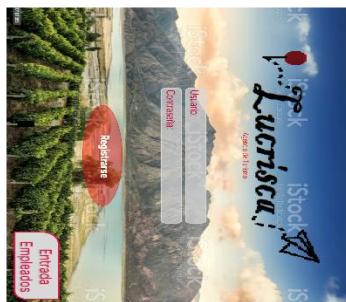
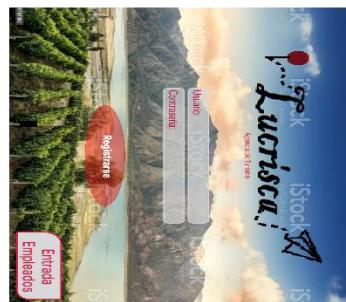
Continuación mostraremos un diagrama de cómo se plantearía nuestra base de datos y una vista previa del sitio web de la agencia.

BASE DE DATOS 2 - LUCRISCA

DIAGRAMA



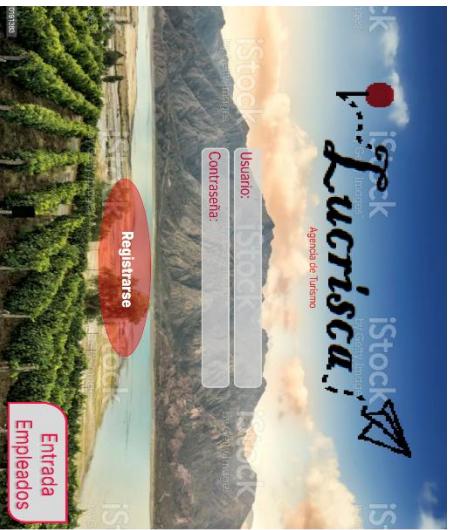
VISTAS







Agenzia di Turismo



REGISTRATE

Nombre:

Apellido:

Fecha de Nacimiento:

Sexo: Hombre Mujer

Teléfono:

Dirección:

Ciudad:

Estado:

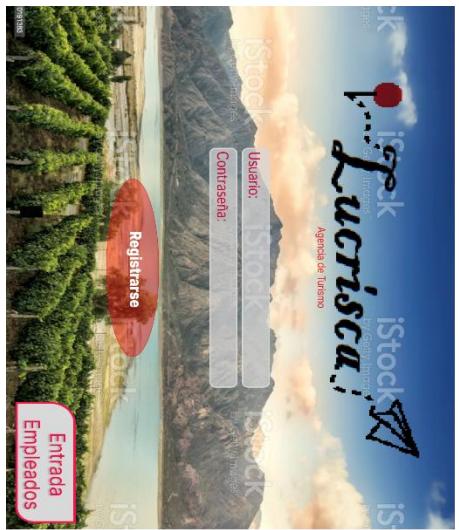
País:

Correo electrónico:

Contraseña:

Confirmar contraseña:

Registrarse



Lucrisca

Autos

Empleados

Rutas

Vehiculos

Facturas



AUTOS

Datos Autos:

Tipo:

Patente:

Capacidad personas:

Año:

Modelo:

Serie:

Permiso:

MODIFICAR



FACTURAS

Datos de la Factura:

Fecha:

Monto:

No de Descuento:

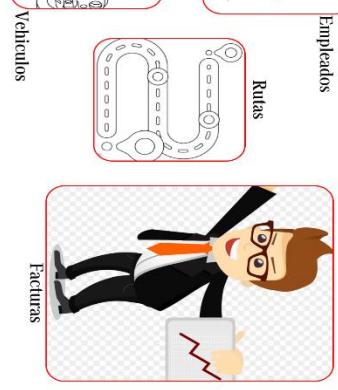
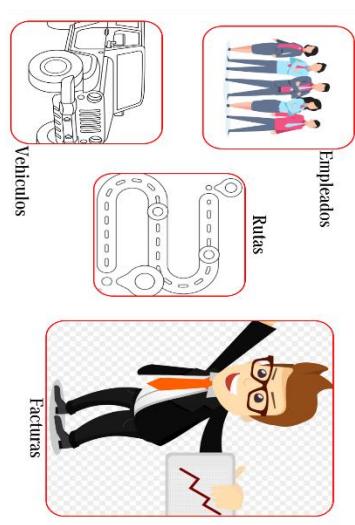
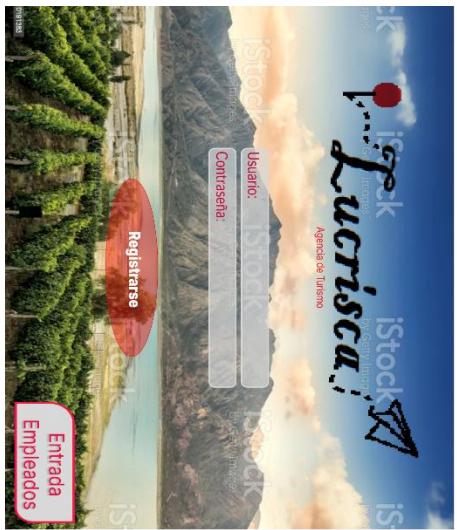
No de Factura:

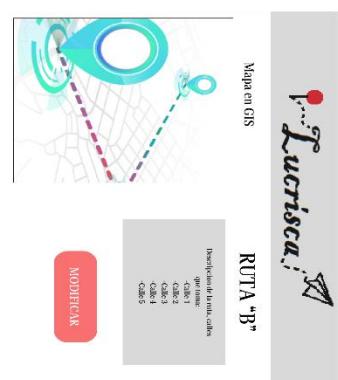
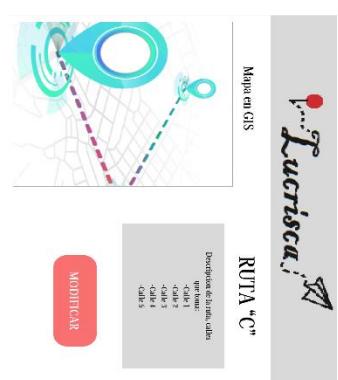
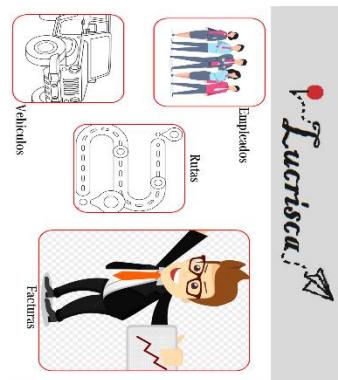
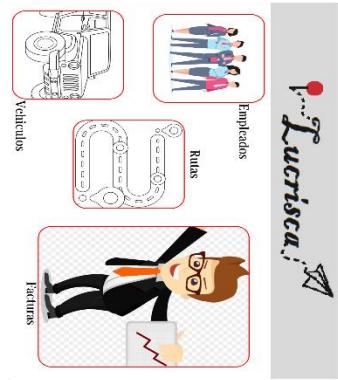
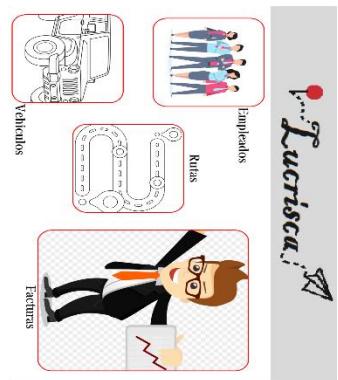
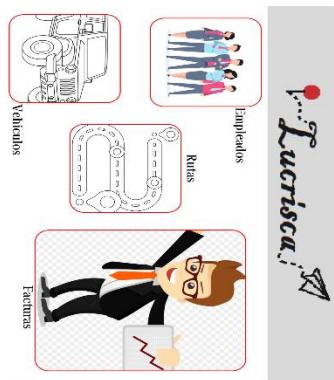
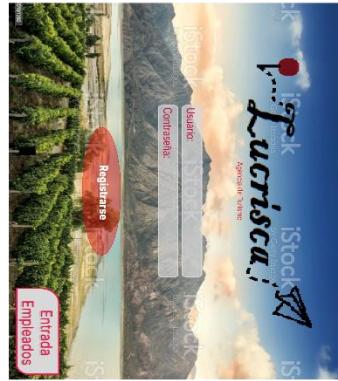
Tipo de Factura:

Costo del Véhic:

Modo de pago:

MODIFICAR







Lucrisca

Lucrisca

Lucrisca

Lucrisca

Stock

Aprovecha

tu tiempo

y

ahorra

dinero

con

nuevas

funciones

y

soluciones

para tu

negocio.

¡Presta

atención

a tus

clientes

y

aumenta

tu

producción

y

reducir

tu

costo

de

operación.

¡Y

no

te

quieres

que

te

quieras

RUTAS

Ahora vamos a mostrar las rutas que ofrece la agencia de viajes, vamos a mostrar los recorridos en formato imagen vectorial del recorrido sobre una imagen satelital, para poder observarlas con detalles, ya que a la hora de verlas en MongoDB sólo veremos sus coordenadas en el GeoJSON.



Ruta 1

Todas las rutas comienzan desde el mismo punto, la Plaza Independencia, y terminan allí. Esta ruta, como pueden ver, da un paseo por el Parque General San Martín, uno de los lugares más importantes y hermosos de la ciudad.

El recorrido continuo hacia el Cerro de la Gloria, hasta la estatua Libertadores, donde hay una pequeña parada para sacarse fotos.

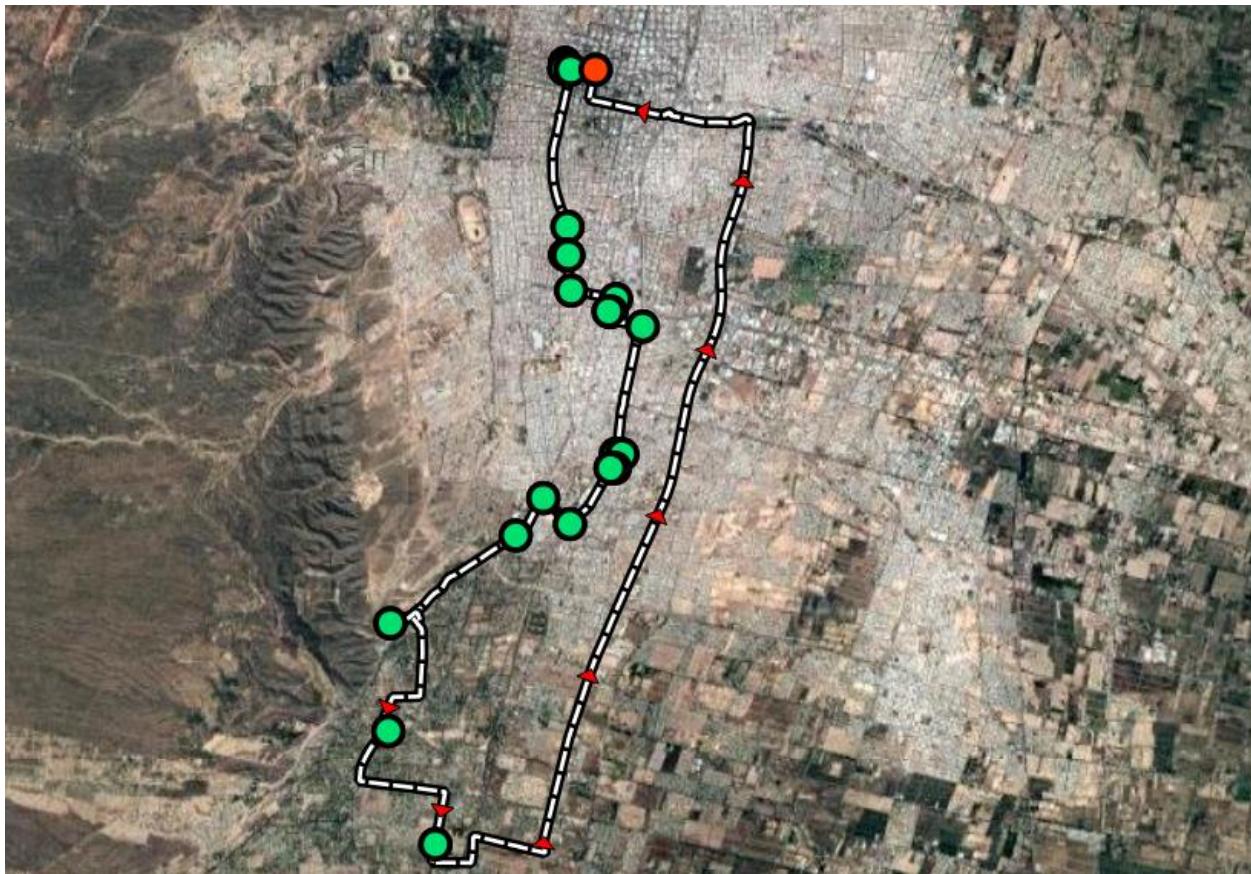
Luego la vuelta será por el parque General San Martín, pero por el otro lado. Es decir, tomara la calle Emilio Civit, y de ahí se dirigirán a su último destino la Plaza Independencia.



Ruta 2

Esta ruta da un recorrido por la Avenida libertadores, pasando por la UNC. Luego sale por Av. Champagnat (que es la que pasa por el Barrio Dalvian) y se dirige hacia la iglesia del Challao. En el lugar se realizará una parada para sacarse fotos y recorrer la zona.

La vuelta hace el mismo recorrido, pero ahora pasa por la Villa Olímpica, y retoma la vuelta por la Av. Libertadores hacia la plaza Independencia donde termina el recorrido.



Ruta 3

Esta ruta comienza en el punto de partida mencionado anteriormente (Plaza Independencia). El recorrido toma la calle Belgrano, transitando varias calles por la Zona de Godoy Cruz, hasta llegar a Palmares Open Mall, donde hay una parada para que puedan recorrer el centro comercial, está parada durara más que las anteriores debido al tamaño del lugar.

Luego arranca el segundo tramo recorriendo (Chacras), pasando por los caracoles de chacras, y finalmente llegando hasta la Plaza de Chacras, donde habrá otra parada.

Luego el tercer tramo, nos lleva desde la Plaza de Chacras hacia una bodega premiada mundialmente en 2007, donde habrá otra para ver el lugar.

Por último, el regreso, se va a realizar por el acceso Sur hasta el acceso Este. Por donde se entra a la ciudad, utilizando la calle Vicente Zapata y luego se arribará a la Plaza Independencia, donde finaliza el recorrido.



Ruta 4

Esta ruta, la más extensa que ofrece la agencia, se dirige al Dique Potrerillos, pasando por la playa de Luján, los caracoles y el túnel de potrerillos. Además, el recorrido hace una pequeñas parada por Cacheuta.

Asimismo, Habrá dos paradas más, una en la Playa de Luján y la otra en el Dique potrerillos. Luego el tramo de regreso será el mismo.

CONSULTAS

Luego de haber creado la BD en MongoDB ahora vamos a realizar algunas consultas para comprobar su funcionamiento y rendimiento.

CONSULTA 1: vamos a medir el tiempo de ejecución al traer o consultar 1000 documentos de la colección clientes.

```
MongoDB Enterprise > db.clientes.find().limit(1000).pretty().explain("executionStats")  
"nReturned" : 1000,  
"executionTimeMillis" : 12,
```

CONSULTA 2: Vamos a consultar ahora entre las primeras 500 reseñas de documentos que tengan 5 estrellas y que hayan comentado "Estuvo muy bueno!!".

```
MongoDB Enterprise > db.feedback.find({"puntuacion": "5", "descripcion": "Estuvo muy bueno!!"}).limit(500).pretty()
```

```
{  
    "_id" : ObjectId("636f1fa20cfcac49e92b0386"),  
    "nrofeedback" : "2127",  
    "DNI" : "3437673",  
    "puntuacion" : "5",  
    "descripcion" : "Estuvo muy bueno!!"  
}  
  
{  
    "_id" : ObjectId("636f1fa20cfcac49e92b0388"),  
    "nrofeedback" : "4805",  
    "DNI" : "2529841",  
    "puntuacion" : "5",  
    "descripcion" : "Estuvo muy bueno!!"  
}  
  
{  
    "_id" : ObjectId("636f1fa20cfcac49e92b0393"),  
    "nrofeedback" : "15685",  
    "DNI" : "4277721",  
    "puntuacion" : "5",  
    "descripcion" : "Estuvo muy bueno!!"  
}  
  
{  
    "_id" : ObjectId("636f1fa20cfcac49e92b0394"),  
    "nrofeedback" : "16172",  
    "DNI" : "333355",  
    "puntuacion" : "5",  
    "descripcion" : "Estuvo muy bueno!!"  
}  
  
{  
    "_id" : ObjectId("636f1fa20cfcac49e92b0399"),  
    "nrofeedback" : "21286",  
    "DNI" : "462313",  
    "puntuacion" : "5",  
    "descripcion" : "Estuvo muy bueno!!"  
}  
  
{  
    "_id" : ObjectId("636f1fa20cfcac49e92b039b"),  
    "nrofeedback" : "23268",  
    "DNI" : "1088481",  
    "puntuacion" : "5",  
    "descripcion" : "Estuvo muy bueno!!"  
}
```

CONSULTA 3: Ahora vamos a consultar los primeros 500 recorridos, donde los buses involucrados tienen 40 L y hayan recorrido al menos 500 km.

```
MongoDB Enterprise > db.kmrecorrido.find({"capacidad": "40", "km": {"$gt": "500"} }).limit(500).pretty()
```

```
{  
    "_id" : ObjectId("636f18f00cfcac49e92af7d1"),  
    "id_kilometraje" : "4178",  
    "capacidad" : "40",  
    "km" : "835"  
}  
{  
    "_id" : ObjectId("636f18f00cfcac49e92af7d9"),  
    "id_kilometraje" : "12655",  
    "capacidad" : "40",  
    "km" : "580"  
}  
{  
    "_id" : ObjectId("636f18f00cfcac49e92af7e5"),  
    "id_kilometraje" : "24478",  
    "capacidad" : "40",  
    "km" : "817"  
}  
{  
    "_id" : ObjectId("636f18f00cfcac49e92af7ed"),  
    "id_kilometraje" : "32439",  
    "capacidad" : "40",  
    "km" : "787"  
}  
{  
    "_id" : ObjectId("636f18f00cfcac49e92af7f1"),  
    "id_kilometraje" : "36376",  
    "capacidad" : "40",  
    "km" : "679"  
}  
{  
    "_id" : ObjectId("636f18f00cfcac49e92af7f2"),  
    "id_kilometraje" : "37895",  
    "capacidad" : "40",  
}
```

CONSULTA 4: Vamos a consultar los primeros 1000 clientes que se llaman Isaac, que su número de documento sea mayor a 1000000 y que se muestren de manera descendente por su apellido.

```
MongoDB Enterprise > db.clientes.find({ "nombre": "Isaac", "DNI": { "$gte": "1000000" } }).limit(1000).sort({ "apellido": -1 }).pretty
```

```
[{"_id": ObjectId("636f0ddf0cfcac49e92aed9b"), "nombre": "Isaac", "apellido": "Young", "direccion": {"calle": "Halton Street", "numero": "616", "piso": "7", "CP": "64309"}, "mail": "isaac.young82@lucrisca.com", "telefono": "35008", "DNI": "238952"}, {"_id": ObjectId("636f0ddf0cfcac49e92af43a"), "nombre": "Isaac", "apellido": "Wilson", "direccion": {"calle": "Avalwood Avenue", "numero": "1162", "piso": "10", "CP": "28312"}, "mail": "isaac.wilson94@lucrisca.com", "telefono": "13957", "DNI": "4085814"}, {"_id": ObjectId("636f0ddf0cfcac49e92aedb1"), "nombre": "Isaac", "apellido": "Walsh", "direccion": {"calle": "Baldwin Ridgeway", "numero": "532", "piso": "19", "CP": "4175541"}]
```

CONSULTA 5: Vamos a consultar los vehículos modelos anteriores a 2010 y que su capacidad de combustible sea entre 50 y 70, también que la capacidad de personas sea menor a 40 y que traiga solo los tipos de vehículo Colectivo

```
Administrator: Símbolo del sistema - mongo
MongoDB Enterprise > db.vehiculos.find({"modelo": {"$lt": 2010}, "capacidad": {"$gte": 50}, "capacidad": {"$lte": 70}, "capacidad_personas": {"$lt": 40}, "tipo_vehiculo": "Colectivo"}).pretty()
{
  "_id" : ObjectId("636fd557f1ad1bd6d132821b"),
  "patente_vehiculo" : "WWW1031",
  "tipo_vehiculo" : "Colectivo",
  "id_kilometraje" : "1495880",
  "capacidad" : "40",
  "capacidad_personas" : 13,
  "marca" : "Jeep",
  "modelo" : 1985,
  "seguro" : 103956503
}

{
  "_id" : ObjectId("636fd557f1ad1bd6d132821d"),
  "patente_vehiculo" : "FFF1051",
  "tipo_vehiculo" : "Colectivo",
  "id_kilometraje" : "2495199",
  "capacidad" : "60",
  "capacidad_personas" : 10,
  "marca" : "Ford",
  "modelo" : 1994,
  "seguro" : 105817465
}
```

CONSULTA 6: Consulta el DNI, junto con el nombre y apellido de aquellos clientes que hayan comprado al menos un viaje y además hayan dejado un feedback de su viaje, de 4 estrellas o más. El cliente debe tener un DNI de entre 20 y 30 millones:

```

db.Cliente.aggregate([
  {
    $lookup: {
      "from": "Feedback",
      "foreignField": "DNI",
      "localField": "DNI",
      "as": "clientes_que_si_dieron_feedback"
    }
  },
  {
    $unwind: "$clientes_que_si_dieron_feedback"
  },
  { $match : { "clientes_que_si_dieron_feedback" :
              { $ne : [] }
            }
  },
  {
    $lookup: {
      "from": "Factura",
      "foreignField": "DNI",
      "localField": "DNI",
      "as": "clientes_que_compraron_viaje"
    }
  },
  {
    $unwind: "$clientes_que_compraron_viaje"
  },
  { $match : { "clientes_que_compraron_viaje" :
              { $ne : [] }
            }
  },
  { $match : { "DNI" :
              { $gte : 2000000, $lte : 3000000
                }
            }
  },
  { $match : { "clientes_que_si_dieron_feedback.puntuacion" :
              { $gte : 4
                }
            }
  },
  {
    $project:{ 
      "DNI": "$DNI",
      "nombre": "$nombre",
      "apellido": "$apellido",
      "clientes_viaje": "$clientes_que_compraron_viaje",
      "clientes_feedback": "$clientes_que_si_dieron_feedback"
    }
  }
])

```

```
[  
  {  
    _id: ObjectId("637136d76aed9e3b4a05914b"),  
    DNI: 2010838,  
    nombre: 'Piers',  
    apellido: 'Henderson',  
    clientes_viaje: {  
      _id: ObjectId("63706ba844530dfad7692a8f"),  
      nrofactura: 14437756,  
      tipo_factura: 'B',  
      DNI: 2010838,  
      fecha: '17/05/2009 08:35 AM',  
      monto_total: 911542,  
      codigo_viaje: 1167556,  
      nro_descuento: 4  
    },  
    clientes_feedback: {  
      _id: ObjectId("636fbc6208601a8b8ecac04f"),  
      nrofeedback: 237364,  
      DNI: 2010838,  
      puntuacion: 5,  
      descripcion: 'Estuvo muy bueno!!'  
    }  
  },  
  {  
    _id: ObjectId("637136d76aed9e3b4a059189"),  
    DNI: 2072211,  
    nombre: 'Dylan',  
    apellido: 'Greene',  
    clientes_viaje: {  
      _id: ObjectId("63706ba844530dfad7692868"),  
      nrofactura: 8928356,  
      tipo_factura: 'B',  
      DNI: 2072211,  
      fecha: '14/04/2003 09:53 PM',  
      monto_total: 529849,  
      codigo_viaje: 1013436,  
      nro_descuento: 4  
    },  
    clientes_feedback: {  
      _id: ObjectId("636fbc6208601a8b8ecac1bf"),  
      nrofeedback: 605757,  
      DNI: 2072211,  
      puntuacion: 5,  
      descripcion: 'Buena atencion'  
    }  
  },  
  {  
    _id: ObjectId("637136d76aed9e3b4a059189"),  
    DNI: 2072211,  
    nombre: 'Dylan',  
    apellido: 'Greene',  
    clientes_viaje: {  
      _id: ObjectId("63706ba844530dfad76929a1"),  
      nrofactura: 12051900,  
      tipo_factura: 'C',  
      DNI: 2072211,  
      fecha: '27/04/2008 12:14 PM',  
      monto_total: 897139,  
      codigo_viaje: 33985,  
      nro_descuento: 1  
    }  
  }]
```

Prueba de rendimiento de la consulta para 10 valores cargados en colección Cliente

DEVOLUCIÓN:

```
executionTimeMillis: 23,
```

Vemos que la consulta demora 23ms en ejecutar:

Pruebo nuevamente con 100 valores:

```
executionTimeMillis: 151,
```

Vemos que la consulta demora 151 ms en ejecutar:

Ahora con 1000 valores:

```
executionTimeMillis: 1308,
```

Vemos que la consulta demora 1308 ms en ejecutar:

Por último 4000 valores:

```
executionTimeMillis: 5689,
```

Vemos que la consulta demora 5689 ms en ejecutar:

Como se puede ver el tiempo de respuesta conforme aumentan datos es demasiado grande, para mejorar esto vamos a ayudarnos, agregando índices al atributo de DNI de las colecciones involucradas:

```
lucriscaprueba@aa> db.Cliente.createIndex({"DNI": 1 })
DNI_1
lucriscaprueba@aa> db.Factura.createIndex({"DNI": 1 })
DNI_1
lucriscaprueba@aa> db.Feedback.createIndex({"DNI": 1 })
DNI_1
```

Consulto nuevamente con 4000 datos, esta vez indexado:

```
executionTimeMillis: 669,
```

Comparación entre antes y después del indexado:

```
executionStats: {  
    executionSuccess: true,  
    nReturned: 3998,  
    executionTimeMillis: 5689,  
    totalKeyExamined: 0,  
    totalDocsExamined: 3998,  
    executionStages: {  
        stage: 'PROJECTION_SIMPLE',  
        nReturned: 3998,  
        executionTimeMillisEstimate: 0,  
        works: 4000,  
        advanced: 3998,  
        needTime: 1,  
        needYield: 0,  
        saveState: 5,  
        restoreState: 5,  
        isEOF: 1,  
        transformBy: {  
            DNI: 1,  
            id: 1  
        }  
    }  
}  
  
ANTES
```

```
executionSuccess: true,  
nReturned: 3998,  
executionTimeMillis: 669,  
totalKeyExamined: 0,  
totalDocsExamined: 3998,  
executionStages: {  
    stage: 'PROJECTION_SIMPLE',  
    nReturned: 3998,  
    executionTimeMillisEstimate: 0,  
    works: 4000,  
    advanced: 3998,  
    needTime: 1,  
    needYield: 0,  
    saveState: 5,  
    restoreState: 5,  
    isEOF: 1,  
    transformBy: {  
        DNI: 1,  
        _id: 1,  
    }  
}  
  
DESPUES
```

CONSULTA 7: Consulta 2: Encontrar en todas las rutas (4 en total), cada uno de los detalles de los empleados encargados de la ruta, junto con una lista de sus vehículos autorizados con sus respectivas coordenadas del trayecto a efectuar y los últimos 4 feedbacks del recorrido.

```
MongoDB Enterprise > db.Ruta.aggregate([
...   {
...     $lookup:{
...       "from": "Vehiculo",
...       "foreignField": "patente_vehiculo",
...       "localField": "patentes_autorizadas.patente_vehiculo",
...       "as": "detalle_vehiculos_autorizados"
...     }
...   },
...   {
...     $unwind: "$detalle_vehiculos_autorizados"
...   },
...   {
...     $lookup:{
...       "from": "Empleado",
...       "foreignField": "id_empleado",
...       "localField": "empleados_involucrados.id_empleado",
...       "as": "detalle_empleados"
...     }
...   },
...
...   {
...     $project:{
...       "vehiculos_autorizadas": "$detalle_vehiculos_autorizados",
...       "empleados": "$detalle_empleados",
...       "mapa": "$mapa.features.geometry.coordinates",
...       "rutanro": "$codigo_ruta",
...       "duracion": "$duracion_estimada_minutos",
...       "puntuacion_media" : "$puntuacion_media" ,
...       "feedbacks": "$ultimos_feedbacks"
...     },
...   }
... ])
```

Devuelve:

```
[  
  {  
    _id: ObjectId("63715d7a53205326b2962036"),  
    vehiculos_autorizadas: {  
      _id: ObjectId("636fbcd508601a8b8ecacf33"),  
      patente_vehiculo: 'AAAA1344',  
      tipo_vehiculo: 'Traffic',  
      id_kilometraje: '560445',  
      capacidad: '40',  
      capacidad_personas: 46,  
      marca: 'Toyota',  
      modelo: 1970,  
      seguro: 134153826  
    },  
    empleados: [  
      {  
        _id: ObjectId("636fbbf508601a8b8ecaafae"),  
        id_empleado: 1090991,  
        nombre: 'Adam',  
        apellido: 'Terry',  
        cuil: '27-1090358-8',  
        mail: 'adam.terry67@lucrisca.com',  
        telefono: 413170,  
        obra_social: 109054316,  
        id_area_empleado: 1,  
        sueldo: 321097,  
        tipo_sangre: 'B-',  
        art: 'True',  
        telefono_emergencia: 596446,  
        licencia: 'A'  
      },  
      {  
        _id: ObjectId("636fbbf508601a8b8ecaafaf"),  
        id_empleado: 1091607,  
        nombre: 'Peter',  
        apellido: 'Bailey',  
        cuil: '27-1091988-8',  
        mail: 'peter.bailey1@lucrisca.com',  
        telefono: 500857,  
        obra_social: 109155948,  
        id_area_empleado: 4,  
        sueldo: 293962,  
        tipo_sangre: 'B+',  
        art: 'True',  
        telefono_emergencia: 145501,  
        licencia: 'A'  
      }  
    ],  
    mapa: [  
      [  
        [  
          [  
            [-68.8458037, -32.8894615],  
            [-68.847059, -32.8892148],  
            [-68.848325, -32.8890002],  
            [-68.849709, -32.888732],  
            [-68.8504815, -32.8886139],  
            [-68.8525414, -32.8882062],  
            [-68.8589787, -32.8870368],  
            [-68.8609529, -32.8866184],  
            [-68.8612318, -32.8865862]  
          ]  
        ]  
      ]  
    ]  
  ]
```

```
[ -68.847059, -32.8892148 ],
[ -68.8458037, -32.8894615 ]
]
]
],
rutano: 1,
duracion: 56,
puntuacion_media: 4.3,
feedbacks: [
{
  nrofeedback: 6580,
  DNI: 437810,
  puntuacion: 5,
  descripcion: 'Quite a lovely trip'
},
{
  nrofeedback: 7505,
  DNI: 268916,
  puntuacion: 5,
  descripcion: 'As good as Lucy!!!'
},
{
  nrofeedback: 8214,
  DNI: 4651649,
  puntuacion: 5,
  descripcion: 'Hermosa experiencia'
},
{
  nrofeedback: 9388,
  DNI: 4022916,
  puntuacion: 5,
  descripcion: 'Muy lindo viajeee'
}
],
{
  _id: ObjectId("63715d7a53205326b2962036"),
  vehiculos_autorizadas: {
    _id: ObjectId("636fbcd508601a8b8ecacf31"),
    patente_vehiculo: 'DDDD1321',
    tipo_vehiculo: 'Colectivo',
    id_kilometraje: '2190196',
    capacidad: '40',
    capacidad_personas: 37,
    marca: 'Citroen',
    modelo: 1997,
    seguro: 132613024
  },
  empleados: [
    {
      _id: ObjectId("636fbbf508601a8b8ecaafae"),
      id_empleado: 1090991,
      nombre: 'Adam',
      apellido: 'Terry',
      cuil: '27-1090358-8',
      mail: 'adam.terry67@lucrisca.com',
      telefono: 413170,
      obra_social: 109054316,
      id_area_empleado: 1,
      sueldo: 321097,
      tipo_sangre: 'B-'
    }
  ]
}
```

Como podemos ver la consulta es demasiado larga para mostrarla toda, por lo cual hemos decidido recortar una parte y mostrar el rendimiento de la misma.

Prueba de rendimiento:

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 156,
  totalKeysExamined: 0,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'PROJECTION_DEFAULT',
    nReturned: 4,
    executionTimeMillisEstimate: 0,
    works: 6,
    advanced: 4,
    needTime: 1,
    needYield: 0,
    saveState: 1,
    restoreState: 1,
    isEOF: 1,
    transformBy: {
      _id: 1,
      codigo_ruta: 1,
      detalle_empleados: 1,
      detalle_vehiculos_autorizados: 1,
      duracion_estimada_minutos: 1,
      'empleados_involucrados.id_empleado': 1,
      'mapa.features.geometry.coordinates': 1,
      'patentes_autorizadas.patente_vehiculo': 1,
      puntuacion_media: 1,
```

Como vemos el tiempo de respuesta es largo, por lo cual agregaremos un índice para así agilizar la consulta. Debemos tener en cuenta que no hace falta hacer uno para DNI, puesto que ya tiene un índice por la consulta anterior.

Creamos el índice:

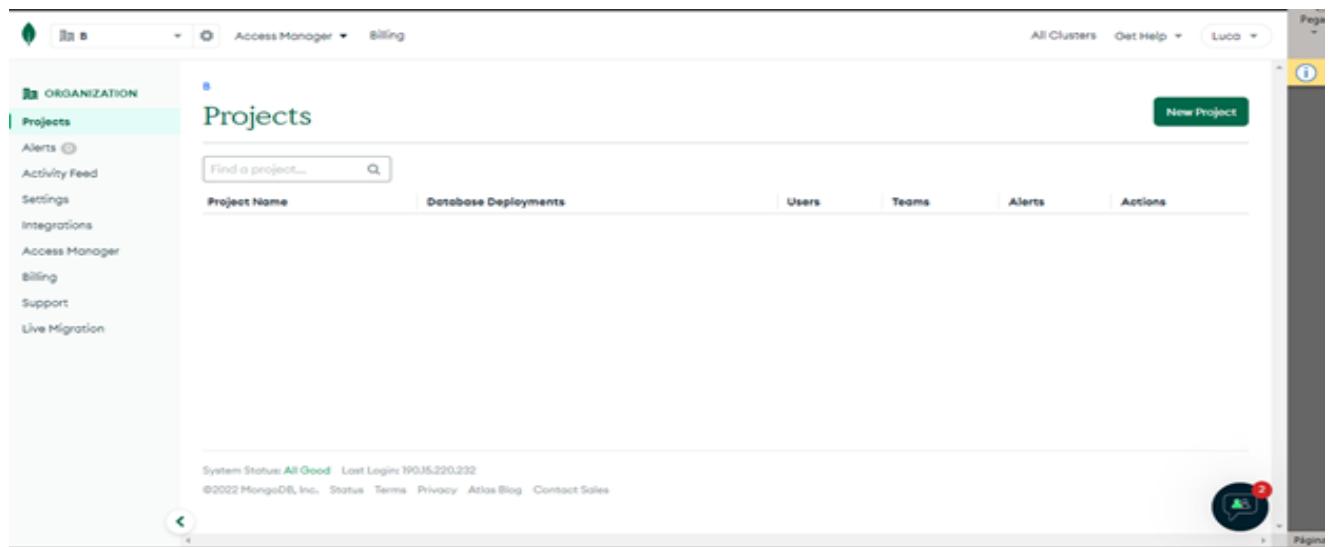
```
switched to db.lucriscapruебaa
lucriscapruебaa> db.Vehiculo.createIndex({"patente_vehiculo":1})
patente_vehiculo_1
lucriscapruебaa> db.Empleado.createIndex({"id_empleado":1})
id_empleado_1
lucriscapruебaa>
```

Se puede observar una leve mejoría:

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 40,
  totalKeysExamined: 0,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'PROJECTION_DEFAULT',
    nReturned: 4,
    executionTimeMillisEstimate: 0,
    works: 6,
    advanced: 4,
    needTime: 1,
    needYield: 0,
    saveState: 1,
    restoreState: 1,
    isEOF: 1,
    transformBy: {
      _id: 1,
      codigo_ruta: 1,
      detalle_empleados: 1,
      detalle_vehiculos_autorizados: 1,
      duracion_estimada_minutos: 1,
      'empleados_involucrados.id_empleado': 1,
      'mapa.features.geometry.coordinates': 1,
      'patentes_autorizadas.patente_vehiculo': 1,
      puntuacion_media: 1,
      ultimos_feedbacks: 1
    }
  }
}
```

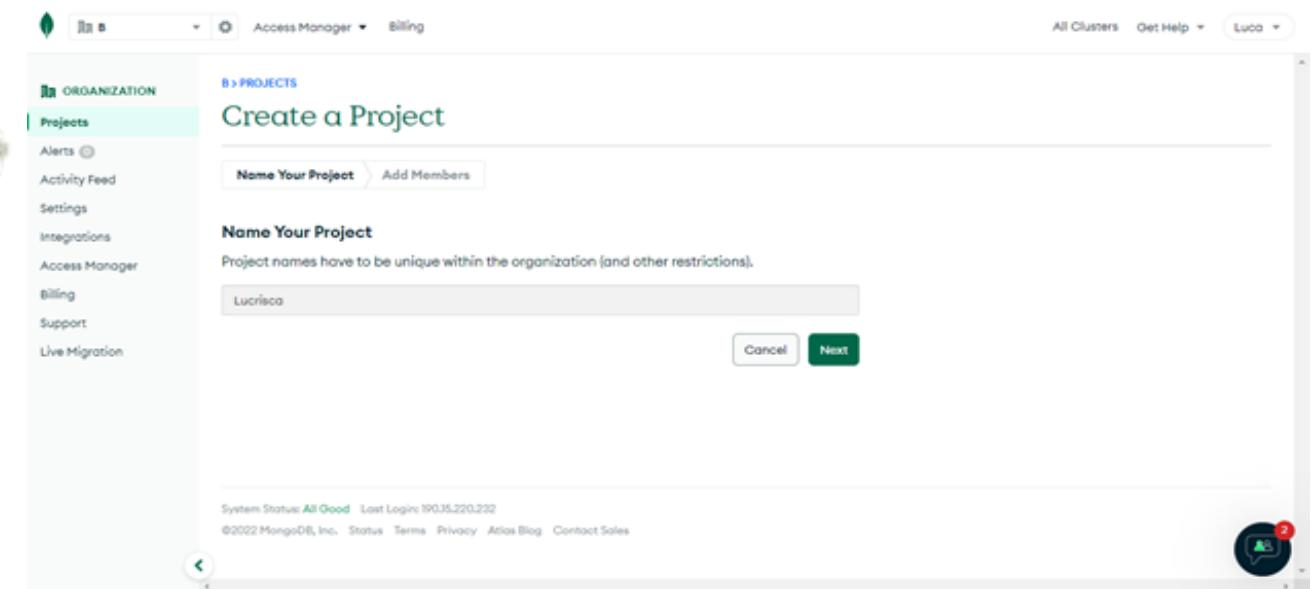
Cloud Mongo Atlas

Ahora crearemos un Cluster, el cuál sirve para ver la base de datos desde cualquier lado. Para crear un cluster, abrimos una cuenta en Mongo Atlas.



The screenshot shows the MongoDB Atlas interface. On the left, there's a sidebar with 'ORGANIZATION' selected, containing links for Projects, Alerts, Activity Feed, Settings, Integrations, Access Manager, Billing, Support, and Live Migration. The main area is titled 'Projects' and shows a table with columns for Project Name, Database Deployments, Users, Teams, Alerts, and Actions. A search bar at the top says 'Find a project...'. At the bottom, it says 'System Status: All Good' and includes links for Lost Logins, MongoDB Inc., Status, Terms, Privacy, Atlas Blog, and Contact Sales. There's also a notification icon with a red '2' in the bottom right corner.

Luego de hacer la cuenta, creamos un nuevo proyecto y añadimos a los miembros que van a poder leer y/o editar el mismo. En este caso todos los integrantes del grupo son propietarios de la DB.



The screenshot shows the 'Create a Project' wizard. The sidebar on the left is identical to the previous screenshot. The main area is titled 'Create a Project' and has two tabs: 'Name Your Project' (selected) and 'Add Members'. Under 'Name Your Project', there's a note: 'Project names have to be unique within the organization (and other restrictions)'. A text input field contains 'Lucrísco'. At the bottom are 'Cancel' and 'Next' buttons. The footer includes 'System Status: All Good', a lost login link, and standard MongoDB links.

The screenshot shows the 'Add Members and Set Permissions' section of a project configuration page. On the left, there's a sidebar with 'ORGANIZATION' and 'Projects' selected. The main area has a heading 'Add Members and Set Permissions' with a search bar. Below it, users can give members access permissions. There are three entries:

- l.teranova@alumno.um.edu.ar (you) - Project Owner
- e.ulbornoz@alumno.um.edu - Project Owner
- e.centelejhe@alumno.um.edu - Project Owner

Buttons at the bottom include 'Cancel', 'Go Back', and 'Create Project'. To the right, a sidebar lists 'Project Member Permissions' with descriptions for each role.

Role	Description
Project Owner	Has full administration access
Project Cluster Manager	Can update clusters
Project Data Access Admin	Can access and modify a cluster's data and indexes, and kill operations
Project Data Access Read/Write	Can access a cluster's data and indexes, and modify data
Project Data Access Read Only	Can access a cluster's data and indexes
Project Search Index Editor	Can view and manage a cluster's search indexes
Project Read Only	May only modify personal

Ahora elegimos el proveedor de la nube, en este caso AWS, y la región donde se va a almacenar.

The screenshot shows the 'Cloud Provider & Region' selection screen. It features a header with 'AWS, Sao Paulo (sa-east-1)' and a sidebar with a 'Cloudy' icon. The main area has tabs for 'Cloud Provider & Region' (selected), 'AWS', 'Google Cloud', and 'Azure'. Below are region selection sections for 'NORTH AMERICA', 'EUROPE', 'AUSTRALIA', 'AFRICA', 'SOUTH AMERICA', and 'MIDDLE EAST'. Each section lists regions with icons and names like 'N. Virginia (us-east-1)', 'Ireland (eu-west-1)', etc. At the bottom, there's a 'FREE' offer note, a 'Back' button, a 'Create Cluster' button, and a sidebar with a 'Lucas' profile.

Seleccionamos el método de autenticación, USUARIO Y CONTRASEÑA.

The screenshot shows the 'Security Quickstart' section of the MongoDB Atlas interface. On the left, a sidebar lists 'DEPLOYMENT' (Database, Data Lake), 'DATA SERVICES' (Triggers, Data API, Data Federation), and 'SECURITY' (Quickstart, Database Access, Network Access, Advanced). Under 'Quickstart', 'New On Atlas' is selected. The main area displays a step titled 'How would you like to authenticate your connection?' with two options: 'Username and Password' (selected) and 'Certificate'. Below this, a detailed description explains creating a database user with a username and password, noting they will have read and write permissions by default. It also mentions updating permissions later. A form field for 'Username' contains 'LucasTerranovaB', and another for 'Password' contains '....'. Buttons for 'Autogenerate Secure Password' and 'Copy' are present. A large green 'Create User' button is at the bottom. The top navigation bar includes 'Access Manager', 'Billing', 'All Clusters', 'Get Help', and 'Lucas'.

Más abajo nos pregunta cómo queremos conectarnos a la DB, en nuestro caso, “My Local Environment”.

The screenshot shows the 'Where would you like to connect from?' section of the MongoDB Atlas interface. The sidebar remains the same as the previous screen. The main area has a checked checkbox next to 'My Local Environment'. A description below it says: 'Use this to add network IP addresses to the IP Access List. [This can be modified at any time.]'. To the right, a 'Cloud Environment' section is shown with a description: 'Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.' Below these sections, a 'Add entries to your IP Access List' section is displayed. It contains a note: 'Only an IP address you add to your Access List will be able to connect to your project's clusters. You can manage existing IP entries via the Network Access Page.' A table allows adding entries with columns 'IP Address' (with input field 'Enter IP Address') and 'Description' (with input field 'Enter description'). A 'Add My Current IP Address' button is also present. At the bottom, a table shows the current 'IP Access List' entry: '190.15.220.232/32' under 'IP Address' and 'My IP Address' under 'Description'. Buttons for 'EDIT' and 'REMOVE' are shown. A 'Finish and Close' button is at the bottom right.

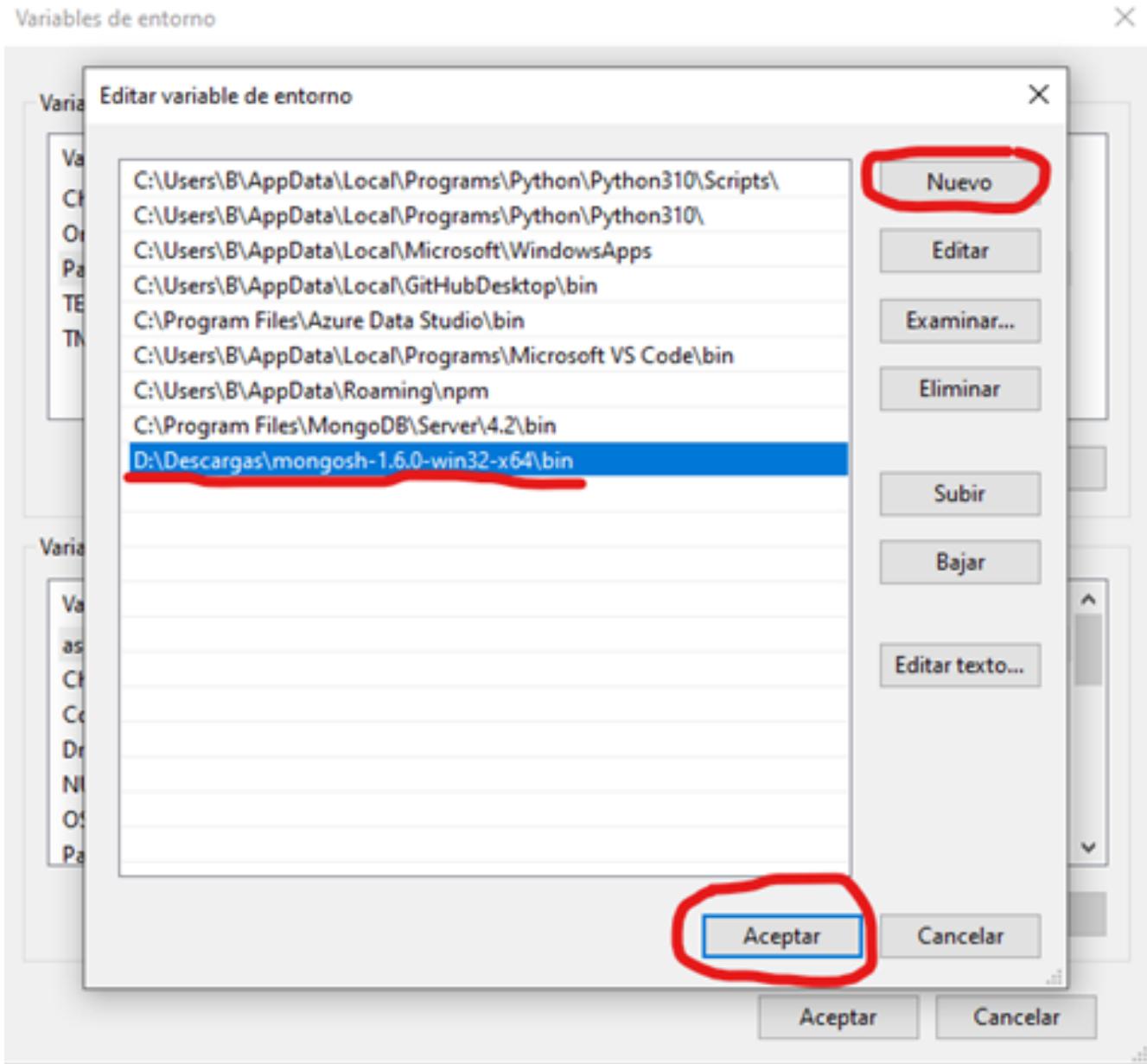
Vemos que el cluster se ha creado exitosamente.

The screenshot shows the MongoDB Atlas interface. On the left, there's a sidebar with options like Deployment, Database (selected), Data Services, Security, and Advanced. The main area is titled "Database Deployments" and shows a single cluster named "Cluster0". The cluster status is "Up and Running". It provides metrics such as R: 0, W: 0, Connections: 0, and Data Size: 0.0 B. Below the metrics, it lists details like Version: 5.0.13, Region: AWS / São Paulo (ap-south-1), Cluster Tier: M0 Sandbox (General), Type: Replica Set - 3 nodes, Backups: Inactive, Linked App Services: None, and Atlas Search: Create Index. A green "Upgrade" button is visible. At the bottom, there are links for System Status, All Databases, Status, Terms, Privacy, Atlas Blog, and Contact Sales.

Elegimos que nos conectaremos al Cluster por medio de MongoDB Shell.

This screenshot shows a modal dialog titled "Connect to Cluster0". It has three tabs at the top: "Setup connection security" (which is active and highlighted in green), "Choose a connection method" (in blue), and "Connect" (in grey). The "Choose a connection method" tab contains a sub-link "View documentation". Below this, a message says "Get your pre-formatted connection string by selecting your tool below." There are four options listed: 1) "Connect with the MongoDB Shell" (with a green icon of a terminal window), which is also highlighted with a red checkmark. 2) "Connect your application" (with a green icon of a gear and dollar sign). 3) "Connect using MongoDB Compass" (with a green icon of a bar chart). 4) "Connect using VS Code" (with a green icon of a code editor). At the bottom of the dialog are "Go Back" and "Close" buttons.

Para conectarnos necesitamos tener instalado mongosh.



Ahora ejecutamos “mongosh” y nos conectamos.

```
mongosh mongodb+srv://<credentials>@cluster0.lfxyvwl.mongodb.net/myFirstDatabase
Microsoft Windows [Versión 10.0.19045.2251]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Windows\system32>mongosh "mongodb+srv://cluster0.lfxyvwl.mongodb.net/myFirstDatabase" --apiVersion 1 --username LucaTerranova8
Enter password: ****
Current Mongosh Log ID: 6372ece7ff8cd667d86c2d55
Connecting to:      mongosh+srv://<credentials>@cluster0.lfxyvwl.mongodb.net/myFirstDatabase?appName=mongosh+1.6.0
Using MongoDB:      5.0.13 (API Version 1)
Using Mongosh:      1.6.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Warning: Found ~/.mongorc.js, but not ~/.mongosrc.js. ~/.mongorc.js will not be loaded.
        You may want to copy or rename ~/.mongorc.js to ~/.mongosrc.js.
Atlas atlas-kckitq-shard-0 [primary] myFirstDatabase>
```

```
mongosh mongodb+srv://<credentials>@cluster0.lfxyvwl.mongodb.net/myFirstDatabase
Atlas atlas-kckitq-shard-0 [primary] myFirstDatabase> "mongodb://localhost:27017"
mongodb://localhost:27017
Atlas atlas-kckitq-shard-0 [primary] myFirstDatabase>
```

BACKUP

Lo siguiente en el proyecto es crear un backup, el cual será el respaldo de nuestra base de datos y definir la política de seguridad.

Para la política elegimos usar **SNAPSHOTS**, porque la recuperación de la misma es muy rápida. Es el método más fácil, simple y eficaz.

Normalmente tienen integración con Windows volumen shadow services (VSS), y es apto para trabajar con un gran volumen de datos. Lo que quiere decir que puede crear a nivel de Windows snapshots antes de tomar el snapshot de la máquina virtual.

Luego el proveedor nos crea una aplicación que es consistente con backups de cualquier aplicación que soporte esta tecnología (VSS).

Por ejemplo, en el caso de AWS los clientes pueden usarlo para proteger EC2, RDS, Redshift, Dynamo entre otros servicios de AWS. Debemos resaltar que en algunos casos el snapshot por nube es la única opción.

Sin embargo, hay que tener en cuenta el tamaño del snapshot ya que el precio por gigabyte puede tornarse muy excesivo.

En nuestro caso no representaría problemas ya que por ahora no es una base de datos tan grande.

Ahora pasaremos a crear un respaldo de nuestra base de datos, debemos seguir los siguientes pasos. Primero abrimos **cmd** con permisos de Administrador.

Ingreso en el Directorio/Ruta donde queremos la exportación, en nuestro caso la insertamos en la carpeta donde se encuentra MongoDB para ubicarla más fácil. Creamos una Subcarpeta llamada Exportaciones.

```
C:\Program Files\MongoDB\Server\4.2\bin\Exportaciones>
```

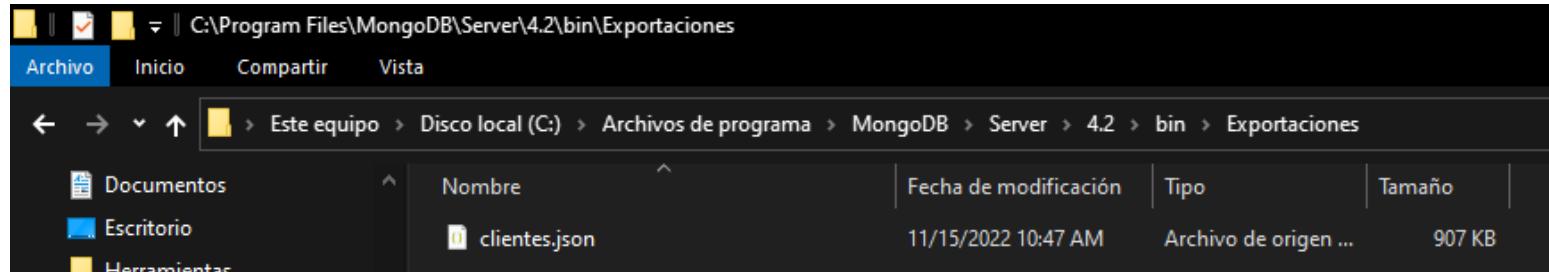
Ejecutar comando para exportar la colección deseada desde la DB en juego.

```
C:\Program Files\MongoDB\Server\4.2\bin\Exportaciones>mongoexport --db Lucrisca --collection clientes --out clientes.json
```

Verificamos la devolución de la consola. Vemos que se han exportado 3998 Documentos de la Colección.

```
C:\Program Files\MongoDB\Server\4.2\bin\Exportaciones>mongoexport --db Lucrisca --collection clientes --out clientes.json
2022-11-15T10:47:26.253-0300    connected to: mongodb://localhost/
2022-11-15T10:47:26.349-0300    exported 3998 records
```

Revisamos que se haya exportado la colección correctamente en el directorio.



Efectivamente se ve en el directorio elegido la colección exportada.

Ahora vamos a hacer el proceso inverso y vamos a importar la colección que exportamos anteriormente, primero tenemos que eliminar la colección.

```
C:\Program Files\MongoDB\Server\4.2\bin\Exportaciones>db.clientes.drop({writeConcern: {w: 1}})
```

Ahora creamos la colección clientes de nuevo e importamos el fichero JSON que habíamos exportado. Podemos observar que los **3998** Documentos han sido importados con éxito.

```
C:\Program Files\MongoDB\Server\4.2\bin\Exportaciones>mongoimport --db Lucrisca --collection clientes --file clientes.json
2022-11-15T11:10:16.160-0300    connected to: mongodb://localhost/
2022-11-15T11:10:16.246-0300    3998 document(s) imported successfully. 0 document(s) failed to import.
```

```
C:\Program Files\MongoDB\Server\4.2\bin\Exportaciones>
```

Ahora vemos que la colección existe y que tiene sus respectivos documentos.

```
MongoDB Enterprise > use Lucrisca
switched to db Lucrisca
MongoDB Enterprise > show collections
area_empleados
clientes
empleados
facturas
feedback
kmrecorrido
promos
rutas
tipo_promo
vehiculos
viajes
MongoDB Enterprise > db.clientes.count()
3998 ←
MongoDB Enterprise >
```

DUMP Y RESTORE

En este apartado decidimos eliminar toda nuestra base de datos y luego restauramos la misma. Para esto realizamos los siguientes pasos.

Abrir **cmd** en modo administrador y seleccionar directorio donde se va a realizar la descarga masiva de datos alojados en nuestro servidor.

```
C:\Windows\system32> cd C:\Users\B\OneDrive\Ingeniería Informática\3er AÑO\DISEÑO DE BASES DE DATOS II\Profesor\Mongodump
```

Ejecutamos comando “mongodump”.

```
C:\Windows\system32> cd C:\Users\B\OneDrive\Ingeniería Informática\3er AÑO\DISEÑO DE BASES DE DATOS II\Profesor\Mongodump
C:\Users\B\OneDrive\Ingeniería Informática\3er AÑO\DISEÑO DE BASES DE DATOS II\Profesor>mongodump
2022-11-15T11:43:56.598-0300      writing admin.system.version to dump\admin\system.version.bson
2022-11-15T11:43:56.605-0300      done dumping admin.system.version (1 document)
2022-11-15T11:43:56.605-0300      writing Lucrisca.clientes to dump\Lucrisca\clientes.bson
2022-11-15T11:43:56.618-0300      done dumping Lucrisca.clientes (3998 documents)
2022-11-15T11:43:56.619-0300      writing Lucrisca.viajes to dump\Lucrisca\viajes.bson
2022-11-15T11:43:56.625-0300      done dumping Lucrisca.viajes (1999 documents)
2022-11-15T11:43:56.626-0300      writing Lucrisca.feedback to dump\Lucrisca\feedback.bson
2022-11-15T11:43:56.630-0300      done dumping Lucrisca.feedback (1000 documents)
2022-11-15T11:43:56.630-0300      writing Lucrisca.vehiculos to dump\Lucrisca\vehiculos.bson
2022-11-15T11:43:56.634-0300      done dumping Lucrisca.vehiculos (899 documents)
2022-11-15T11:43:56.635-0300      writing Lucrisca.rutas to dump\Lucrisca\rutas.bson
2022-11-15T11:43:56.638-0300      done dumping Lucrisca.rutas (4 documents)
2022-11-15T11:43:56.639-0300      writing Lucrisca.tipo_promo to dump\Lucrisca\tipo_promo.bson
2022-11-15T11:43:56.641-0300      done dumping Lucrisca.tipo_promo (4 documents)
2022-11-15T11:43:56.642-0300      writing Lucrisca.promos to dump\Lucrisca\promos.bson
2022-11-15T11:43:56.643-0300      done dumping Lucrisca.promos (4 documents)
2022-11-15T11:43:56.644-0300      writing Lucrisca.area_empleados to dump\Lucrisca\area_empleados.bson
2022-11-15T11:43:56.645-0300      done dumping Lucrisca.area_empleados (4 documents)
2022-11-15T11:43:56.915-0300      writing Lucrisca.facturas to dump\Lucrisca\facturas.bson
2022-11-15T11:43:56.917-0300      writing Lucrisca.empleados to dump\Lucrisca\empleados.bson
2022-11-15T11:43:56.927-0300      done dumping Lucrisca.facturas (2000 documents)
2022-11-15T11:43:56.929-0300      writing Lucrisca.kmrecorrido to dump\Lucrisca\kmrecorrido.bson
2022-11-15T11:43:56.935-0300      done dumping Lucrisca.empleados (3000 documents)
2022-11-15T11:43:56.940-0300      done dumping Lucrisca.kmrecorrido (2999 documents)

C:\Users\B\OneDrive\Ingeniería Informática\3er AÑO\DISEÑO DE BASES DE DATOS II\Profesor>
```

Ahora verificamos en el directorio que se hayan creado las respectivas carpetas.

Nombre	Fecha de modificación	Tipo	Tamaño
admin	11/15/2022 11:43 AM	Carpetas de archivos	
Lucrisca	11/15/2022 11:43 AM	Carpetas de archivos	

Corroboramos la existencia de la carpeta admin.

Nombre	Fecha de modificación	Tipo	Tamaño
system.version.bson	11/15/2022 11:43 AM	Archivo BSON	1 KB
system.version.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB

Corroboramos la existencia de la carpeta de la BD (Lucrisca).

Nombre	Fecha de modificación	Tipo	Tamaño
area_empleados.bson	11/15/2022 11:43 AM	Archivo BSON	1 KB
area_empleados.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
clientes.bson	11/15/2022 11:43 AM	Archivo BSON	864 KB
clientes.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
empleados.bson	11/15/2022 11:43 AM	Archivo BSON	900 KB
empleados.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
facturas.bson	11/15/2022 11:43 AM	Archivo BSON	332 KB
facturas.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
feedback.bson	11/15/2022 11:43 AM	Archivo BSON	114 KB
feedback.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
kmrecorrido.bson	11/15/2022 11:43 AM	Archivo BSON	234 KB
kmrecorrido.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
promos.bson	11/15/2022 11:43 AM	Archivo BSON	1 KB
promos.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
rutas.bson	11/15/2022 11:43 AM	Archivo BSON	94 KB
rutas.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
tipo_promo.bson	11/15/2022 11:43 AM	Archivo BSON	1 KB
tipo_promo.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
vehiculos.bson	11/15/2022 11:43 AM	Archivo BSON	180 KB
vehiculos.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB
viajes.bson	11/15/2022 11:43 AM	Archivo BSON	211 KB
viajes.metadata.json	11/15/2022 11:43 AM	Archivo de origen ...	1 KB

Ahora restauraremos la base de datos, para lo cual abriremos el **cmd** en modo admin y entramos al directorio donde se encuentra el backup realizado por “mongodump”

```
C:\Users\B\OneDrive\Ingeniería Informática\3er AÑO\DISEÑO DE BASES DE DATOS II\Profesor\Mongodump\dump\Lucrisca>mongorestore /host:localhost /port:27017 /db:Lucrisca /dir:"./"
```

Vamos a ver que tira error debido a que estamos intentando restaurar una BD que ya tenemos.

```
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718cb') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718cc') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718cd') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718ce') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718cf') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718d0') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718d1') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718d2') }
2022-11-15T12:10:51.346-0300 continuing through error: E11000 duplicate key error collection: Lucrisca.viajes index: _id_ dup key: { _id: ObjectId('6370ba6a5beeffa2434718d3') }
2022-11-15T12:10:51.346-0300 no indexes to restore
2022-11-15T12:10:51.346-0300 finished restoring Lucrisca.viajes (0 documents, 1999 failures)
2022-11-15T12:10:51.346-0300 0 document(s) restored successfully. 15911 document(s) failed to restore.

C:\Users\B\OneDrive\Ingeniería Informática\3er AÑO\DISEÑO DE BASES DE DATOS II\Profesor\Mongodump\dump\Lucrisca>
```

Ahora vamos a eliminar la base de datos así podemos recuperarla.

```
Administrator: Símbolo del sistema - mongo
MongoDB Enterprise > use dbs
switched to db dbs
MongoDB Enterprise > db.dropDatabase()
{ "ok" : 1 }
MongoDB Enterprise > show dbs
Lucrisca  0.002GB
admin      0.000GB
config     0.000GB
local      0.000GB
MongoDB Enterprise > use Lucrisca
switched to db Lucrisca
MongoDB Enterprise > db.dropDatabase()
{ "dropped" : "Lucrisca", "ok" : 1 }
MongoDB Enterprise > show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
MongoDB Enterprise >
```

Ahora vemos que la DB ha sido restaurada con éxito, solo que ya no tenemos los índices, los tendremos que volver a agregar. Finalizando así el Restore.

```
2022-11-15T12:17:52.283-0300    reading metadata for Lucrisca.clientes from clientes.metadata.json
2022-11-15T12:17:52.283-0300    reading metadata for Lucrisca.facturas from facturas.metadata.json
2022-11-15T12:17:52.283-0300    reading metadata for Lucrisca.kmrecorrido from kmrecorrido.metadata.json
2022-11-15T12:17:52.301-0300    restoring Lucrisca.facturas from facturas.bson
2022-11-15T12:17:52.346-0300    no indexes to restore
2022-11-15T12:17:52.346-0300    finished restoring Lucrisca.facturas (2000 documents, 0 failures)
2022-11-15T12:17:52.347-0300    reading metadata for Lucrisca.viajes from viajes.metadata.json
2022-11-15T12:17:52.360-0300    restoring Lucrisca.viajes from viajes.bson
2022-11-15T12:17:52.399-0300    no indexes to restore
2022-11-15T12:17:52.399-0300    finished restoring Lucrisca.viajes (1999 documents, 0 failures)
2022-11-15T12:17:52.399-0300    reading metadata for Lucrisca.vehiculos from vehiculos.metadata.json
2022-11-15T12:17:52.411-0300    restoring Lucrisca.vehiculos from vehiculos.bson
2022-11-15T12:17:52.436-0300    no indexes to restore
2022-11-15T12:17:52.436-0300    finished restoring Lucrisca.vehiculos (899 documents, 0 failures)
2022-11-15T12:17:52.436-0300    reading metadata for Lucrisca.feedback from feedback.metadata.json
2022-11-15T12:17:52.449-0300    restoring Lucrisca.feedback from feedback.bson
2022-11-15T12:17:52.473-0300    no indexes to restore
2022-11-15T12:17:52.473-0300    finished restoring Lucrisca.feedback (1000 documents, 0 failures)
2022-11-15T12:17:52.474-0300    reading metadata for Lucrisca.rutas from rutas.metadata.json
2022-11-15T12:17:52.484-0300    restoring Lucrisca.rutas from rutas.bson
2022-11-15T12:17:52.488-0300    no indexes to restore
2022-11-15T12:17:52.488-0300    finished restoring Lucrisca.rutas (4 documents, 0 failures)
2022-11-15T12:17:52.488-0300    reading metadata for Lucrisca.promos from promos.metadata.json
2022-11-15T12:17:52.500-0300    restoring Lucrisca.promos from promos.bson
2022-11-15T12:17:52.503-0300    no indexes to restore
2022-11-15T12:17:52.503-0300    finished restoring Lucrisca.promos (4 documents, 0 failures)
2022-11-15T12:17:52.504-0300    reading metadata for Lucrisca.tipo_promo from tipo_promo.metadata.json
2022-11-15T12:17:52.515-0300    restoring Lucrisca.tipo_promo from tipo_promo.bson
2022-11-15T12:17:52.518-0300    no indexes to restore
2022-11-15T12:17:52.518-0300    finished restoring Lucrisca.tipo_promo (4 documents, 0 failures)
2022-11-15T12:17:52.518-0300    reading metadata for Lucrisca.area_empleados from area_empleados.metadata.json
2022-11-15T12:17:52.530-0300    restoring Lucrisca.area_empleados from area_empleados.bson
2022-11-15T12:17:52.533-0300    no indexes to restore
2022-11-15T12:17:52.533-0300    finished restoring Lucrisca.area_empleados (4 documents, 0 failures)
2022-11-15T12:17:52.608-0300    restoring Lucrisca.empleados from empleados.bson
2022-11-15T12:17:52.613-0300    restoring Lucrisca.clientes from clientes.bson
2022-11-15T12:17:52.617-0300    restoring Lucrisca.kmrecorrido from kmrecorrido.bson
2022-11-15T12:17:52.678-0300    no indexes to restore
2022-11-15T12:17:52.678-0300    finished restoring Lucrisca.kmrecorrido (2999 documents, 0 failures)
2022-11-15T12:17:52.690-0300    no indexes to restore
2022-11-15T12:17:52.690-0300    finished restoring Lucrisca.empleados (3000 documents, 0 failures)
2022-11-15T12:17:52.705-0300    no indexes to restore
2022-11-15T12:17:52.705-0300    finished restoring Lucrisca.clientes (3998 documents, 0 failures)
2022-11-15T12:17:52.705-0300    15911 document(s) restored successfully. 0 document(s) failed to restore.
```

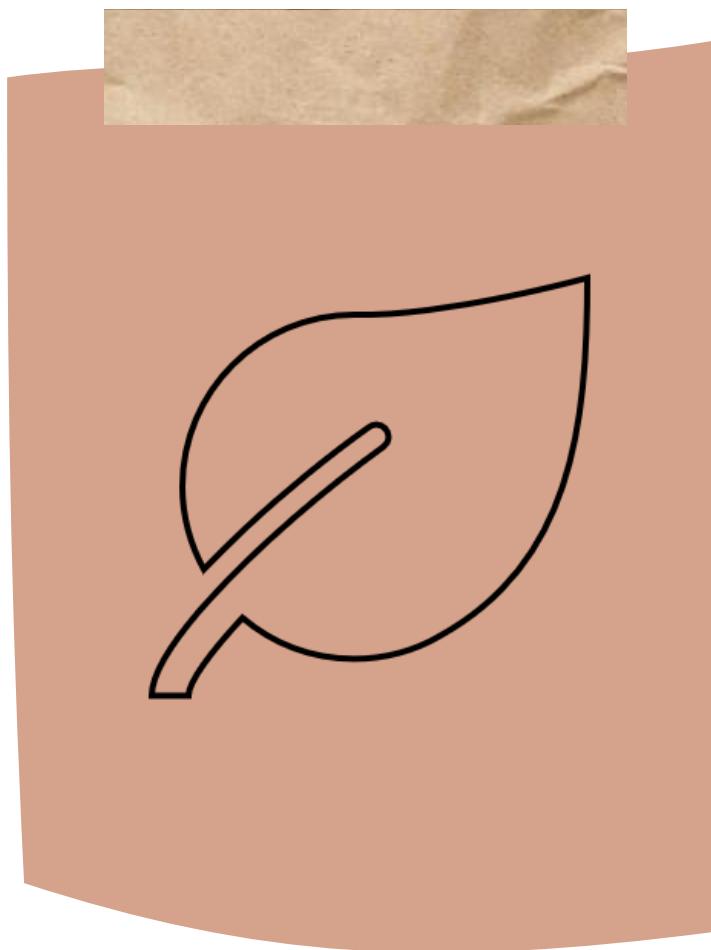
CONCLUSIÓN

En fin, las bases de datos ayudan a la gestión ordenada de la información y a disponer un repositorio de datos que será utilizado principalmente en las aplicaciones del negocio.

Dependiendo del propósito de estas aplicaciones y del tipo de información a almacenar, utilizamos BBDD SQL para las aplicaciones que necesitan datos estructurados y NoSQL para aplicaciones que tienen que manejar volúmenes muy grandes de datos estructurados y no estructurados como las redes sociales.

Al finalizar este proyecto, podemos observar que es beneficioso utilizar la base de datos NOTSQL para designios como nuestra agencia de viajes. Ya que está pensada para la Big Data y orientada a la rapidez del programa, atreves de la utilización de redundancias.

Además, hemos aprendido la importancia de utilizar GeoJSON, cuando debemos mapear rutas.



LINK

❖ **REPOSITORIO:** <https://github.com/LucaTerranovaB/DBII-LUCRISCA>

INTEGRANTES

❖ Integrantes del Grupo:

- ✓ Luca Terranova
- ✓ Cristian Albornoz
- ✓ Carla S. Centeleghe