

Práctica 1

PRÁCTICA 1 SISTEMAS DE INFORMACIÓN. CURSO 2023-2024 GESTIÓN DE BASES DE DATOS NOSQL (CASSANDRA VS MONGODB) Y GENERACIÓN DE REPORTES CON IREPOTS.

Índice

PRÁCTICA 1
SISTEMAS DE INFORMACIÓN.
CURSO 2023-2024
GESTIÓN DE BASES DE DATOS
NOSQL (CASSANDRA VS
MONGODB) Y GENERACIÓN DE
REPORTES CON IREPOTS.

Planificación

Enunciado

Objetivos

Planificación general de tareas

Diagrama Gantt

Diseño

Implementación

Bibliografía

Planificación

Enunciado

Sistema de Gestión de Agencia de Viajes con Cassandra y MongoDB.

Una agencia de viajes necesita un sistema de gestión eficiente para manejar su amplia oferta de destinos, paquetes turísticos y reservas de clientes. Se pide diseñar e implementar un sistema de gestión de agencia de viajes utilizando una base de datos NoSQL tanto con el modelo Cassandra como el modelo MongoDB. El sistema debe ser capaz de realizar consultas básicas, avanzadas y complejas de manera eficiente, lo que implica la generación de índices o tablas con diferentes indexaciones.

Objetivos

1. Instalación y configuración

▼ Cassandra. Mi imagen de cassandra: `docker pull jloen/cassandrasinf:2`

- He usado Docker para la instalación de la imagen de Cassandra `docker pull cassandra`

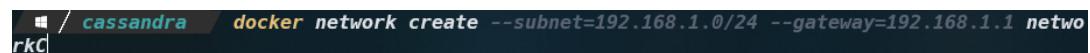


```
Windows PowerShell
cassandra docker pull cassandra
```

- Comprobar que la imagen haya sido descargada correctamente:

<code>docker image ls</code>				
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<code>mongo</code>	<code>latest</code>	<code>5acb2131d51f</code>	<code>8 days ago</code>	<code>757MB</code>
<code>cassandra</code>	<code><none></code>	<code>60f97b970450</code>	<code>4 months ago</code>	<code>355MB</code>
<code>cassandra</code>	<code>latest</code>	<code>597c6621fd1d</code>	<code>4 months ago</code>	<code>355MB</code>

- Creación de red `docker network create --subnet=192.168.1.0/24 --gateway=192.168.1.1 networkC`



```
root@DESKTOP-1D9F5B: ~ / cassandra docker network create --subnet=192.168.1.0/24 --gateway=192.168.1.1 networkC
```

- Comprobar que la red haya sido creada correctamente `docker network ls`

```
■ / cassandra docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
66cf0a751760    bridge    bridge      local
85331389874b    host      host       local
433379e9d71b    miRed     bridge      local
8f4ed6fd5754    networkC  bridge      local
62c2b12b9a86    none      null       local
```

- Creación y arranque del contenedor `docker run --name cassandra1 --network networkC -d -p9042:9042 --ip 192.168.1.94 cassandra:latest`

```
■ / cassandra docker run --name cassandra1 --network networkC -d -p9042:9042 --ip 192.168.1.94 cassandra:latest
```

- Comprobar que el contenedor haya sido creado correctamente `docker ps -a --format="Nombre:\t{{.Names}}\nID:\t{{.ID}}"`

```
■ / cassandra docker ps -a --format="Nombre:\t{{.Names}}\nID:\t{{.ID}}"
Nombre: mongodb1
ID: 22498f67c458
Nombre: mongo
ID: db13741feedb
Nombre: cassandra1
ID: 9b9fd8a48b45
```

- Entrar en la consola de cassandra `docker run -it --network networkC --rm -v .\cqlsh_credentials.txt:/root/.cassandra/cqlshrc cassandra cqlsh 192.168.1.94 9042 cassandra1`

```
■ / cassandra docker run -it --network networkC --rm -v .\cqlsh_credentials.txt:/root/.cassandra/cqlshrc cassandra cqlsh 192.168.1.94 9042 cassandra1

Warning: Password is found in an insecure cqlshrc file. The file is owned or readable by other users on the system.
Notice: Credentials in the cqlshrc file is deprecated and will be ignored in the future.
Please use a credentials file to specify the username and password.

Connected to Test Cluster at 192.168.1.94:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
jloen@cqlsh> |
```

- `.\cqlsh_credentials.txt`: Archivo creado con las credenciales de acceso a la base de datos:

```
■ / cassandra cat .\cqlsh_credentials.txt
[authentication]
username = jloen
password = jloen19
```

- Crear usuario:

Configuración de autenticación:

- Edita el archivo de configuración de Cassandra (`cassandra.yaml`) para habilitar la autenticación.
- Busca la línea `authenticator: AllowAllAuthenticator` y cámbiala a `PasswordAuthenticator` y en caso de que quieras crear roles y asignarlos a tu usuario también tendrías que cambiar la línea `authorizer: AllowAllAuthorizer` y cambiar a `authorizer: CassandraAuthorizer`

```
# /cassandra docker exec -it cassandra1 bash
root@9b9fd8a48b45:/# cd etc/
root@9b9fd8a48b45:/etc# dir
aduser.conf      fstab      ld.so.conf.d  pam.conf    selinux
alternatives     gal.conf   legal        pam.d      shadow
apt              group      libaudit.conf  passwd     shadow-
bash.bashrc      group-    locale.alias  passwd-    shells
bindresvport.blacklist gshadow    locale.gen   profile    skel
ca-certificates  gshadow-   localtime   profile.d  ssl
ca-certificates.conf  gss       login.defs  python3   subgid
cassandra        host.conf  logrotate.d  python3.8 subuid
cloud            hostname  lsb-release  rc0.d     sysctl.conf
cron.d           hosts     machine-id  rc1.d     sysctl.d
cron.daily        init.d    mailcap     rc2.d     systemd
debconf.conf     inputrc   mailcap.order  rc3.d     terminfo
debian_version   iproute2  mime.types   rc4.d     timezone
default          issue    mke2fs.conf  rc5.d     ucf.conf
deluser.conf     issue.net  mtab        rc6.d     update-motd.d
dpkg             kernel    networks   rcS.d     wgetrc
e2scrub.conf    ldap      nsswitch.conf resolv.conf xattr.conf
environment      ld.so.cache opt        rmt
fonts            ld.so.conf  os-release  security
root@9b9fd8a48b45:/etc# cd cassandra/
root@9b9fd8a48b45:/etc/cassandra# cat cassandra.yaml

# Cassandra storage config YAML

# NOTE:
# See https://cassandra.apache.org/doc/latest/configuration/ for
# full explanations of configuration directives
# /NOTE
```

- Desde la shell no es posible modificar el archivo, para ello primero he tenido que copiar dicho archivo a mi máquina local, modificarlo y para luego pegarlo nuevamente a su ubicación original.

```

[cassandra] docker cp cassandra1:/etc/cassandra/cassandra.yaml c:\Users\JLoel\Desktop\cassandra
Successfully copied 90.1kB to c:\Users\JLoel\Desktop\cassandra
[cassandra] ls .\cassandra.yaml
Directory: C:\Users\JLoel\Desktop\cassandra

Mode LastWriteTime Length Name
---- ----- ----- 
-a-- 10/12/2023   16:05    88389  E  cassandra.yaml

```

/ **cassandra** / vim .\cassandra.yaml

```

# - AllowAllAuthenticator performs no checks - set it to disable authentication.
# - PasswordAuthenticator relies on username/password pairs to authenticate
#   users. It keeps usernames and hashed passwords in system_auth.roles table.
#   Please increase system_auth keyspace replication factor if you use this authenticator.
#   If using PasswordAuthenticator, CassandraRoleManager must also be used (see below)
authenticator: PasswordAuthenticator

# Authorization backend, implementing IAuthorizer; used to limit access/provide permissions
# Out of the box, Cassandra provides org.apache.cassandra.auth.{AllowAllAuthorizer,
# CassandraAuthorizer}.
#
# - AllowAllAuthorizer allows any action to any user - set it to disable authorization.
# - CassandraAuthorizer stores permissions in system_auth.role_permissions table. Please
#   increase system_auth keyspace replication factor if you use this authorizer.
authorizer: CassandraAuthorizer

```

- Luego pegar el archivo modificado a la shell del contenedor `docker cp .\cassandra.yaml cassandra1:/etc/cassandra/cassandra.yaml`

/ **cassandra** / docker cp .\cassandra.yaml cassandra1:/etc/cassandra/cassandra.yaml

- Luego reiniciamos el contenedor `docker restart cassandra1`

/ **cassandra** / docker restart cassandra1

- Finalmente procedemos a la creación del usuario `CREATE USER jloen WITH PASSWORD 'jloen19' SUPERUSER;`

jloen@cqlsh> CREATE USER jloen WITH PASSWORD 'jloen19' SUPERUSER;

▼ MongoDB. Mi imagen de mongoDB:

- He usado Docker para la instalación de Mongo `docker pull mongo`

/ **cassandra** / docker pull mongo

- Comprobar que la imagen haya sido descargada correctamente `docker image ls`

```
■ / mongodb docker image ls
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
mongo          latest   5acb2131d51f  8 days ago  757MB
```

- Creación de red `docker network create --subnet=192.168.1.0/24 --gateway=192.168.1.1 networkC`

```
■ / cassandra docker network create --subnet=192.168.1.0/24 --gateway=192.168.1.1 networkC
```

- Comprobar que la red haya sido creada correctamente `docker network ls`

```
■ / cassandra docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
66cf0a751760    bridge    bridge     local
85331389874b    host      host      local
433379e9d71b    miRed    bridge     local
8f4ed6fd5754    networkC  bridge     local
62c2b12b9a86    none      null      local
```

- Creación y arranque del contenedor `docker run --name mongodb --network networkC -d -p97017:97017 --ip 192.168.1.94 mongo`

```
■ / cassandra docker run --name mongodb --network networkC -d -p97017:97017 --ip 192.168.1.94 mongo
```

- Comprobar que el contenedor haya sido creado correctamente `docker ps -a --format="Nombre:\t{{.Names}}\nID:\t{{.ID}}"`

```
■ / cassandra docker ps -a --format="Nombre:\t{{.Names}}\nID:\t{{.ID}}"
Nombre: mongodb1
ID: 22498f67c458
Nombre: mongodb
ID: db13741feedb
Nombre: cassandra1
ID: 9b9fd8a48b45
```

- Entrar en la consola de mongodb `docker exec -it mongodb1 mongosh`

```

# /cassandra docker exec -it mongoDB1 mongosh
Current Mongosh Log ID: 6575dc54303faf9a10f42a05
Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.1.0
Using MongoDB: 7.0.4
Using Mongosh: 2.1.0

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

-----
The server generated these startup warnings when booting
2023-12-10T14:48:48.497+00:00: Using the XFS filesystem is strongly recommended with the
WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2023-12-10T14:48:49.556+00:00: Access control is not enabled for the database. Read and write
access to data and configuration is unrestricted
2023-12-10T14:48:49.559+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We
suggest setting it to 'never'
2023-12-10T14:48:49.559+00:00: vm.max_map_count is too low
-----

test> |

```

- Crear usuario

```

test> use admin
switched to db admin
admin> db.createUser({
...   user: "jloen",
...   pwd: "jloen19",
...   roles: ["userAdminAnyDatabase"],
...   authenticationRestrictions: [
...     {
...       clientSource: ["192.168.1.94"]
...     }
...   ]
... })

```

```

db.createUser({
  user: "jloen",
  pwd: "jloen19",
  roles: ["userAdminAnyDatabase"],
  authenticationRestrictions: [
    {
      clientSource: ["192.168.1.94"]
    }
  ]
})

```

- Comprobar que el usuario se haya creado correctamente `db.getUsers()`

```

admin> db.getUsers()
{
  users: [
    {
      _id: 'admin.jloen',
      userId: UUID('ea428bae-ee9f-4874-8852-e49ebf354439'),
      user: 'jloen',
      db: 'admin',
      roles: [ { role: 'userAdminAnyDatabase', db: 'admin' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}

```

2. **Creación de tablas:** Crear las tablas destinos, paquetes, clientes y reservas con las columnas adecuadas y las claves primarias correspondientes.

- Modelo de Datos.

El modelo de datos constará de las siguientes tablas:

Tabla Destinos.

- Tabla Destinos.

destino_id (UUID): Identificador único del destino (clave primaria).	nombre (texto): Nombre del destino.
pais (texto): País del destino.	descripcion (texto): Descripción del destino
clima (texto): Clima del destino	

- Tabla paquetes.

paquete_id (UUID): Identificador único del paquete (clave primaria).	nombre (texto): Nombre del paquete
destino_id (UUID): Identificador del destino asociado al paquete	duracion (int): Duración del paquete en días
precio (decimal): Precio del paquete	

- Tabla Clientes.

cliente_id (UUID): Identificador único del cliente (clave primaria)	nombre (texto): Nombre del cliente
correo_electronico (texto): Correo electrónico del cliente	telefono (texto): Número de teléfono

cliente.

del cliente

- Tabla Reservas.

reserva_id (UUID): Identificador único de la reserva (clave primaria)	paquete_id (UUID): Identificador único del paquete reservado.
cliente_id (UUID): Identificador único del cliente que realiza la reserva.	fecha_inicio (fecha): Fecha de inicio de la reserva
fecha_fin (fecha): Fecha de finalización de la reserva	pagado (boolean): Indicador de si la reserva ha sido pagada

- Inserción de datos: Insertar al menos 50 destinos, 100 paquetes, 200 clientes y 500 reservas ficticias en sus respectivas tablas.

▼ IMPORTANT

- **Los nombres y correos de los clientes son generados mediante una función de la librería JavaFaker.**
- **Los nombres de los clientes, destinos, descripción de los destinos, paquetes, países y climas proceden del archivo src/main/resources/data/bd.txt**
- Realizar las consultas básicas, avanzadas y complejas mencionadas y verificar los resultados obtenidos.
 - *Consultas Básicas:*
 1. Obtener todos los destinos disponibles en la agencia de viajes.
 2. Obtener los detalles de un paquete turístico específico a través de su ID.
 3. Obtener todas las reservas realizadas por un cliente específico.
 4. Obtener todos los paquetes turísticos disponibles para un destino en particular.
 5. Obtener todos los clientes que han realizado reservas en un rango de fechas específico.
 - *Consultas Avanzadas con Indexación:*
 1. Generar un índice para acelerar la búsqueda de paquetes por nombre.

2. Crear una tabla de resumen para almacenar el número total de reservas realizadas por cada paquete.
 3. Obtener todos los clientes que han realizado reservas en destinos con un clima específico.
 4. Generar un índice compuesto para optimizar la búsqueda de reservas de un cliente en un rango de fechas determinado.
 5. Crear una tabla de índice para acelerar la búsqueda de destinos por país.
- Consultas Complejas con Indexación:
 1. Obtener todos los destinos populares (los más reservados) en orden descendente según el número de reservas.
 2. Generar un índice para acelerar la búsqueda de clientes por correo electrónico.
 3. Crear una tabla de índice para almacenar información sobre la disponibilidad de paquetes por destino y fecha.
 4. Obtener todos los paquetes turísticos disponibles para un destino específico y con una duración determinada.
 5. Generar un índice compuesto para optimizar la búsqueda de reservas por cliente, destino y estado de pago
 - Generar los índices o tablas necesarios para optimizar el rendimiento de las consultas avanzadas y complejas.

3. Conexión a base de datos:

NOTA: Para la conexión a la base de datos modifica el archivo `src/main/resources/mongodb-config.properties` con tus credenciales.

```

mongodbcfg.properties X
1 # mongodb Configuration
2 mongodb.host=192.168.1.92
3 mongodb.port=27017
4 mongodb.username=jloen
5 mongodb.password=jloen19
6 mongodb.database=jloenbd

```

Planificación general de tareas

Fecha	Tarea	Tiempo Estimado	Tiempo empleado
30/11/2023	Entender el enunciado	40m	37m
30/11/2023	Instalar Docker	15m	10m
02/12/2023	Instalar y configurar Cassandra	2h	4h
06/12/2023	Instalar y configurar MongoDB	2h, 20m	3h, 10m
30/11/2023	Entender el funcionamiento y uso de comandos de Docker	1h y 30m	2h, 35m
02/11/2023	Entender el DDL, DML, DCL y DQL en Cassandra	2h y 35m	5h
07/11/2023	Entender el DDL, DML, DCL y DQL en MongoDB	2h y 35m	5h
03/11/2023	Cassandra with Java	3h y 30m	12h
07/11/2023	MongoDB with Java	2h y 20m	8h
10/11/2023	Aprender a generar reportes	3h	4h

Diagrama Gantt

Tiempo_PlanTareas_Practica1_Sinf

PlanTareasPractica1Sinf, <sin consulta>

Informe de parte de horas

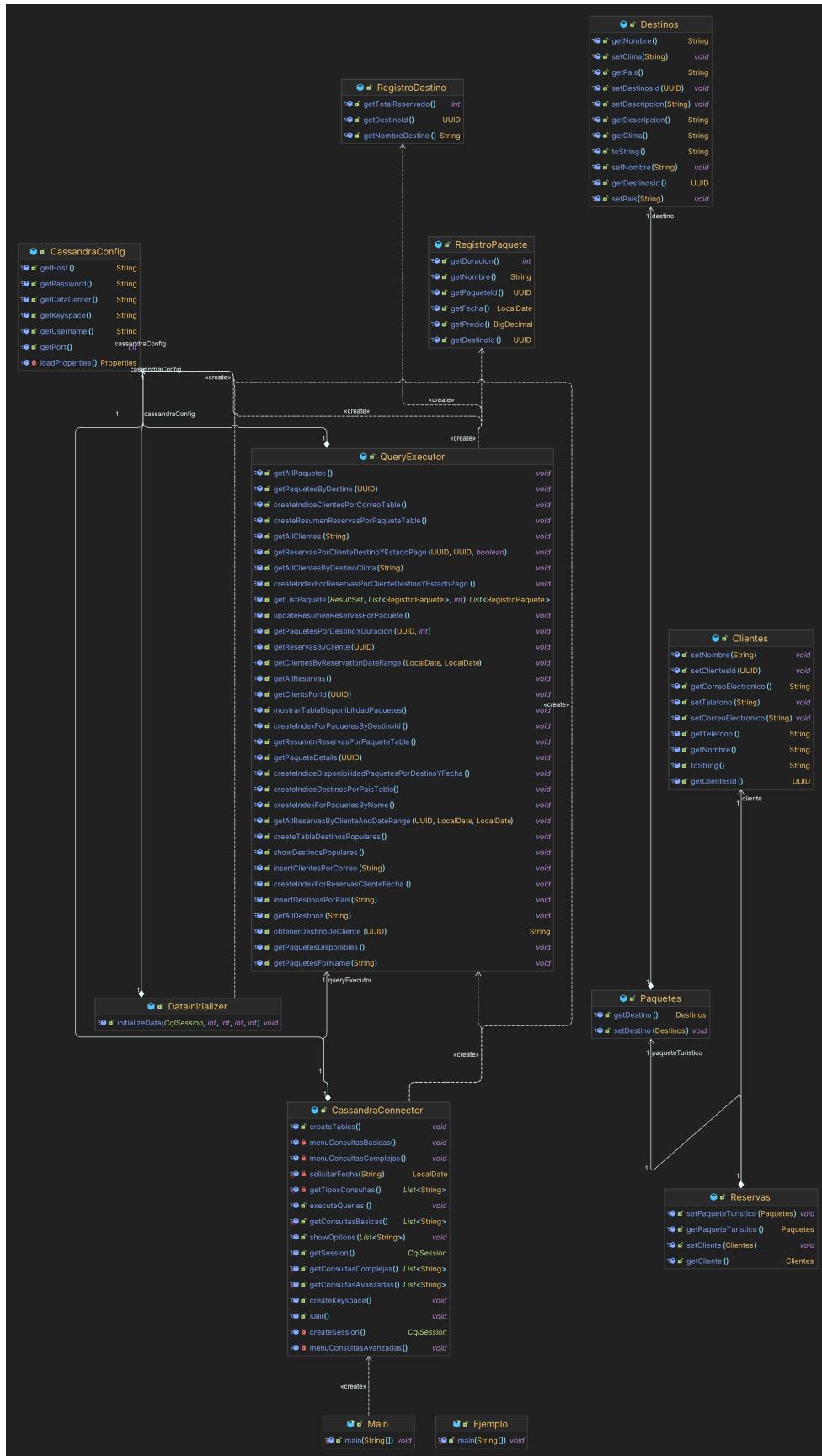
[por usuario](#) [por incidencia](#)

		nov. 2023										
		30 jue.	1 vie.	2 sáb.	3 dom.	4 lun.	5 mar.	6 mié.	7 jue.	8 vie.	9 sáb.	10 dom.
Incidencias												
Tiempo total	55h 12m	13h 12m	00m	9h 00m	12h 00m	00m	00m	4h 00m	13h 00m	00m	4h 00m	
PTP1S-1 Entender el enunciado	37m	37m										
PTP1S-2 Instalar Docker	10h 00m	10h 00m										
PTP1S-3 Instalar y configurar Cassandra	4h 00m		4h 00m									
PTP1S-4 Instalar y configurar MongoDB	4h 00m					4h 00m						
PTP1S-5 Entender el funcionamiento y uso de comandos ...	2h 35m	2h 35m										
PTP1S-6 Entender el DDL, DML, DCL y DQL en Cassandra	5h 00m		5h 00m									
PTP1S-7 Entender el DDL, DML, DCL y DQL en MongoDB	5h 00m					5h 00m						
PTP1S-8 Cassandra with Java	12h 00m				12h 00m							
PTP1S-9 MongoDB with Java	8h 00m					8h 00m						
PTP1S-10 Aprender a generar reportes	4h 00m									4h 00m		

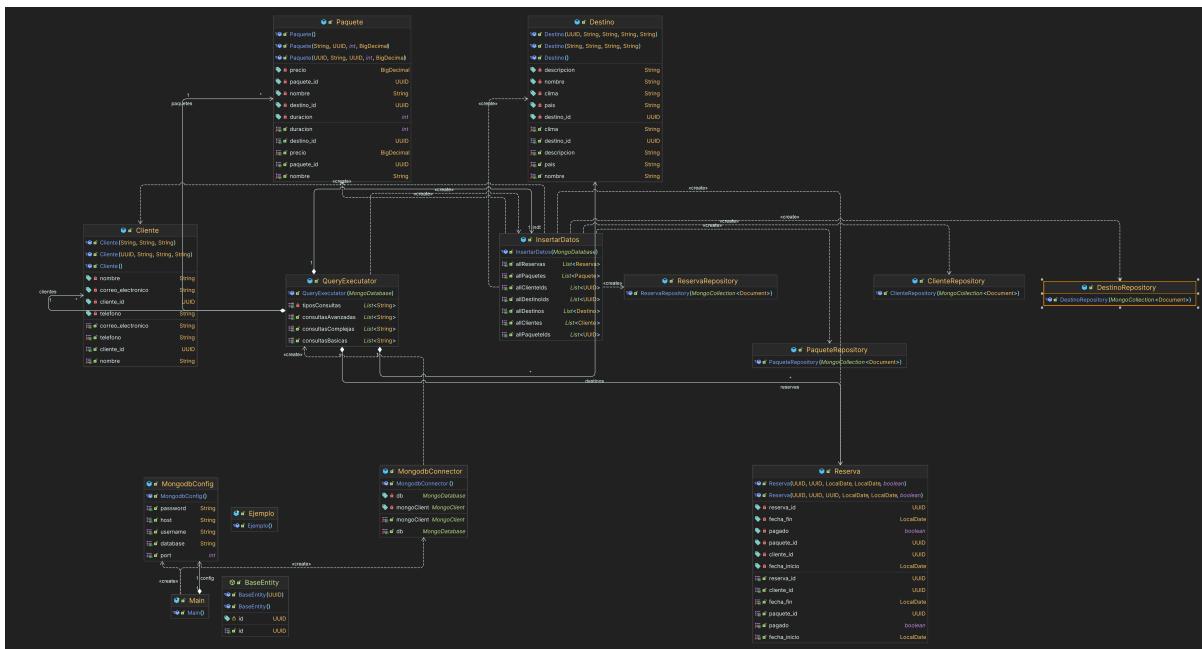
Diseño

1. Diagrama Clase

- Cassandra



- MongoDB



2. Tablas

- Mongodb

```
jloen@bd> show collections
clientes
destinos
disponibilidad_paquetes
paquetes
reservas
resumen_reservas
resumenReservasPorPaquete
temp_destinos
```

- Cassandra

```
jloen@cqlsh:cassandra> describe tables;
clientes          destinos_populares      paquetes
clientes_por_correo  destinos_por_pais    reservas
destinos           disponibilidad_paquetes resumen_reservas
                           resumenReservasPorPaquete
```

Implementación

[Github Cassandra](#)

[Github MongoDB](#)

[READMEActualizado](#)

Mi imagen de cassandra: `docker pull jloen/cassandrasinf:2`

Mi imagen de mongoDB: `docker pull jloen/mongodbsinf:2`

Bibliografía

1. Información sobre Docker: [hostinger.com.ar](#) [bing.com](#) [hostinger.es](#) [kinsta.com](#) [unir.net](#) [openwebinars.net](#) [es.wikipedia.org](#) [ionos.es](#) [es.wikipedia.org](#) [chatgpt](#)



1. Instalación de Docker: [dockerInstall](#)
2. Curso Docker: [Docker Full Curse for Beginners](#) , [Udemy](#) , [Docker and Kubernetes](#)
3. Documentación de la asignatura.
4. [Docker hub](#) para la instalación de cassandra y mongoDB
5. [Chatgpt](#)