UNIVERSIDAD
DE GRANADA

TEMA 4

Trabajo tema 4

Diseño y Desarrollo de Sistemas de Información

GIIMADE - 2024/2025

Introducción

Apache Cassandra es un sistema de gestión de bases de datos NoSQL distribuido y altamente escalable, diseñado para manejar grandes volúmenes de datos en entornos distribuidos con alta disponibilidad. Utiliza un modelo de datos basado en columnas y proporciona replicación y tolerancia a fallos automáticos.

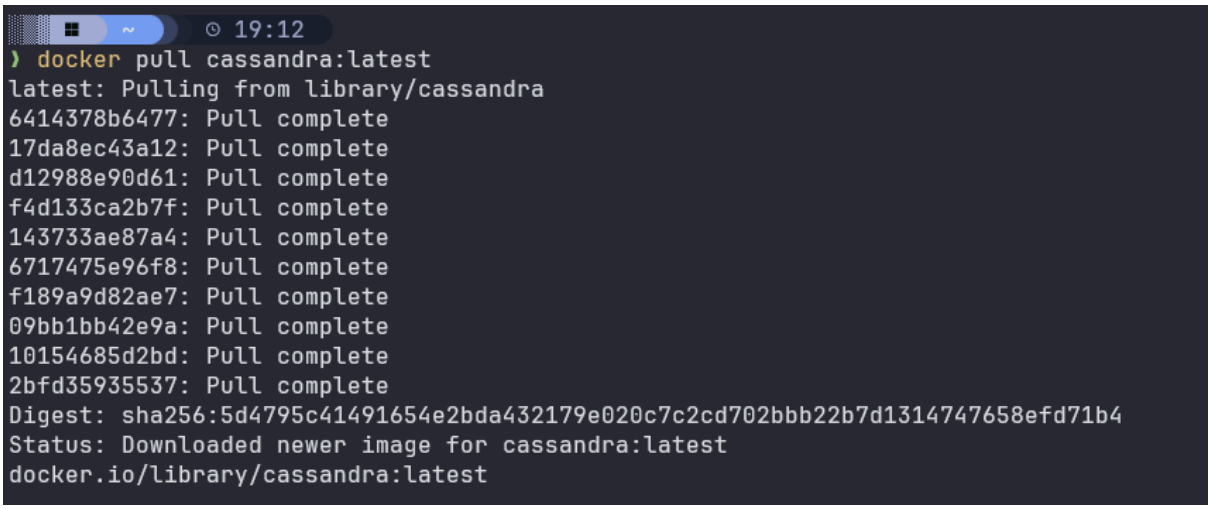
Características clave:

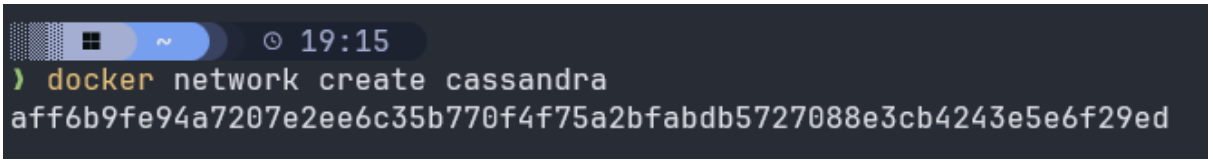
- Modelo de datos basado en columnas (Column Family Store).
- Escalabilidad horizontal.
- Replicación configurable para tolerancia a fallos.
- Consistencia eventual o estricta, configurable.
- Lenguaje de consulta: Cassandra Query Language (CQL).

Instalación de Apache Cassandra

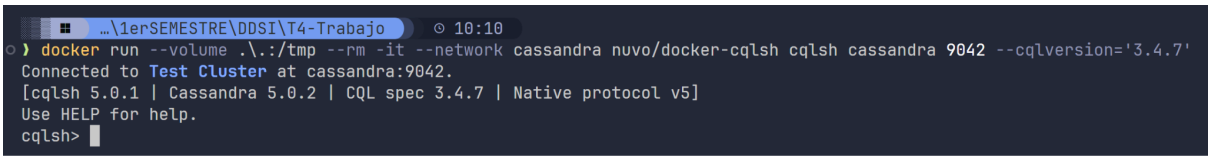
Para instalar Cassandra hemos seguido los pasos descritos en la página oficial: https://cassandra.apache.org/_/quickstart.html

Tendremos que tener instalado Docker Desktop en nuestro ordenador y así, acceder a los puertos del host sin exponerlos en el host.





Para realizar la conexión a la base de datos usaremos la herramienta cqlsh, que es una consola para trabajar con el lenguaje propio de Cassandra CQL. Para su instalación también usaremos Docker.



Sentencias Básicas en CQL

Data definition language (DDL)

1. Hemos creado un Keyspace, equivalente a una base de datos en SQL

```
Unset
CREATE KEYSPACE Hotel
   WITH replication = {
      'class': 'SimpleStrategy',
      'replication_factor': 1
   };
```

2. Después creamos una tabla dentro del Keyspace

```
Unset
 CREATE TABLE Hotel.Clientes (
      dni TEXT PRIMARY KEY,
      nombre TEXT,
      telefono TEXT
   );
 CREATE TABLE Hotel.Habitaciones (
      habitacion_id INT PRIMARY KEY,
      nombre TEXT,
      telefono TEXT
   );
 CREATE TABLE Hotel.Reservas (
      reserva_id UUID PRIMARY KEY,
       dni TEXT,
       habitacion_id INT,
   );
```

Data Manipulation Language (DML)

1. Insertamos datos

Unset

```
INSERT INTO Hotel.Clientes (dni, nombre, telefono)
VALUES ('12345678J', 'Juan', '+34600123456');

INSERT INTO Hotel.Clientes (dni, nombre, telefono)
VALUES ('12345678A', 'Ana', '+34600123457');

INSERT INTO Hotel.Clientes (dni, nombre, telefono)
VALUES ('12345678C', 'Carlos', '+34600123458');

INSERT INTO Hotel.Habitaciones (habitacion_id, capacidad, precio_por_noche)
VALUES (101, 1, 50.00);

INSERT INTO Hotel.Habitaciones (habitacion_id, capacidad, precio_por_noche)
VALUES (102, 2, 75.00);

INSERT INTO Hotel.Habitaciones (habitacion_id, capacidad, precio_por_noche)
VALUES (201, 4, 150.00);

-- Reserva para Juan Pérez en la habitación 101
INSERT INTO Hotel.Reservas (reserva_id, dni, habitacion_id, fecha_inicio,
fecha_fin, estado)
VALUES (uuid(),
        '12345678J',
        101,
        '2025-01-15',
        '2025-01-18',
        'Confirmada');

-- Reserva para Ana López en la habitación 102
INSERT INTO Hotel.Reservas (reserva_id, dni, habitacion_id, fecha_inicio,
fecha_fin, estado)
VALUES (uuid(),
        '12345678A',
        102,
        '2025-02-01',
        '2025-02-05',
        'Pendiente');

-- Reserva para Carlos García en la habitación 201
INSERT INTO Hotel.Reservas (reserva_id, dni, habitacion_id, fecha_inicio,
fecha_fin, estado)
VALUES (uuid(),
        '12345678C',
        201,
        '2025-03-10',
        '2025-03-15',
        'Cancelada');
```

2. Consultamos datos

```
Unset
-- Mostrar contenido de la tabla Clientes
SELECT * FROM Hotel.Clientes;

-- Mostrar contenido de la tabla Habitaciones
SELECT * FROM Hotel.Habitaciones;

-- Mostrar contenido de la tabla Reservas
SELECT * FROM Hotel.Reservas;
```

```
cqlsh> SOURCE '/tmp/GetAllTables.cql'

 dni       | nombre | telefono
-----------+--------+--------------
 12345678C | Carlos | +34600123458
 12345678J |   Juan | +34600123456
 12345678A |    Ana | +34600123457

(3 rows)

 habitacion_id | capacidad | precio_por_noche
---------------+-----------+------------------
           201 |         4 |           150.00
           102 |         2 |            75.00
           101 |         1 |            50.00

(3 rows)

 reserva_id                           | dni       | estado     | fecha_fin  | fecha_inicio | habitacion_id
--------------------------------------+-----------+------------+------------+--------------+---------------
 ad5ee88c-f11f-4962-80e4-8b709a8d3fd2 | 12345678A |  Pendiente | 2025-02-05 |   2025-02-01 |           102
 17040e18-d678-486e-873a-04c1e68431b7 | 12345678J | Confirmada | 2025-01-18 |   2025-01-15 |           101
 8b42d1e4-6150-48bb-b157-34786d2c7541 | 12345678C |  Cancelada | 2025-03-15 |   2025-03-10 |           201

(3 rows)
```

```
Unset
-- Consultar reservas de un cliente específico, por ejemplo, con DNI
'12345678J'
SELECT *
FROM Hotel.Reservas
WHERE dni = '12345678J' ALLOW FILTERING;
```

```
cqlsh> SOURCE '/tmp/GetReserveByDNI.cql'

 reserva_id                           | dni       | estado     | fecha_fin  | fecha_inicio | habitacion_id
--------------------------------------+-----------+------------+------------+--------------+---------------
 17040e18-d678-486e-873a-04c1e68431b7 | 12345678J | Confirmada | 2025-01-18 |   2025-01-15 |           101

(1 rows)
```

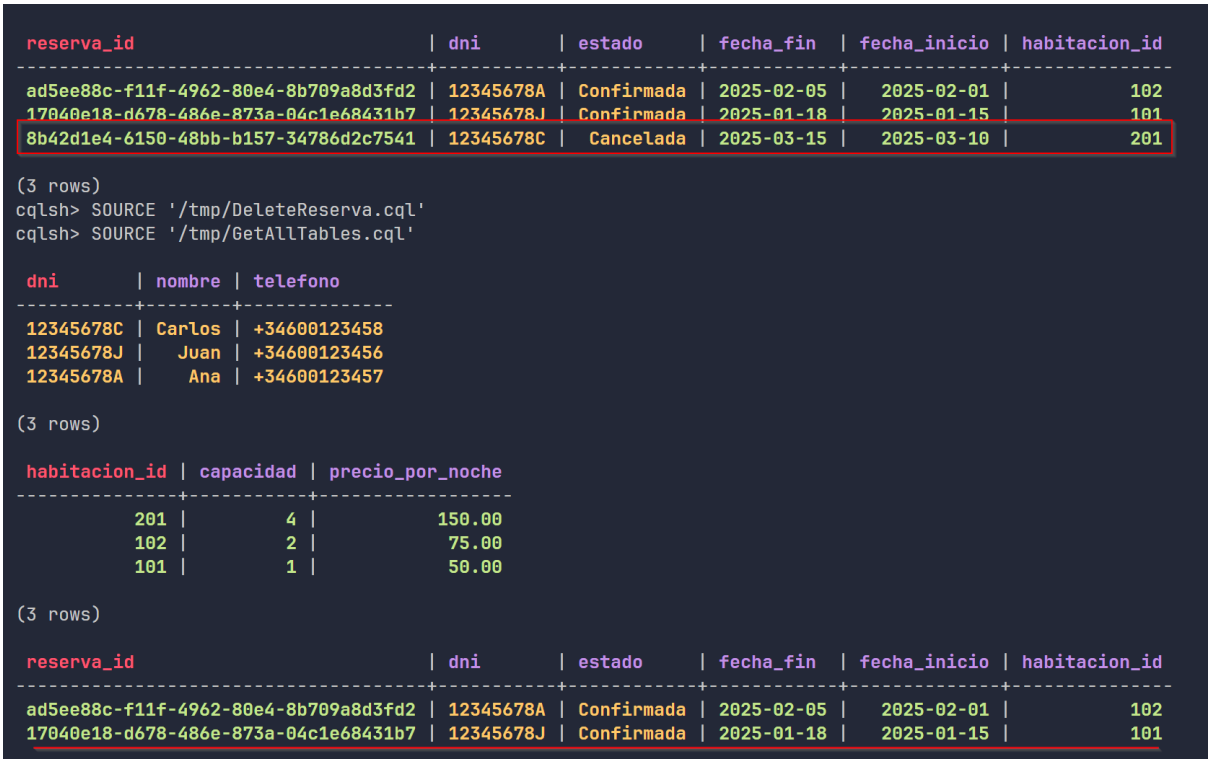
3. Actualizamos datos

```
Unset
-- Modificar el estado de una reserva con un ID específico
UPDATE Hotel.Reservas
SET estado = 'Confirmada'
WHERE reserva_id = ad5ee88c-f11f-4962-80e4-8b709a8d3fd2;
```

```
 reserva_id                           | dni       | estado     | fecha_fin  | fecha_inicio | habitacion_id
--------------------------------------+-----------+------------+------------+--------------+---------------
 ad5ee88c-f11f-4962-80e4-8b709a8d3fd2 | 12345678A |  Pendiente | 2025-02-05 |   2025-02-01 |           102
 17040e18-d678-486e-873a-04c1e68431b7 | 12345678J | Confirmada | 2025-01-18 |   2025-01-15 |           101
 8b42d1e4-6150-48bb-b157-34786d2c7541 | 12345678C |  Cancelada | 2025-03-15 |   2025-03-10 |           201

(3 rows)
cqlsh> SOURCE '/tmp/UpdateEstadoReserva.cql'
cqlsh> SOURCE '/tmp/GetAllTables.cql'

 dni       | nombre | telefono
-----------+--------+--------------
 12345678C | Carlos | +34600123458
 12345678J |   Juan | +34600123456
 12345678A |    Ana | +34600123457

(3 rows)

 habitacion_id | capacidad | precio_por_noche
---------------+-----------+------------------
           201 |         4 |           150.00
           102 |         2 |            75.00
           101 |         1 |            50.00

(3 rows)

 reserva_id                           | dni       | estado     | fecha_fin  | fecha_inicio | habitacion_id
--------------------------------------+-----------+------------+------------+--------------+---------------
 ad5ee88c-f11f-4962-80e4-8b709a8d3fd2 | 12345678A | Confirmada | 2025-02-05 |   2025-02-01 |           102
 17040e18-d678-486e-873a-04c1e68431b7 | 12345678J | Confirmada | 2025-01-18 |   2025-01-15 |           101
 8b42d1e4-6150-48bb-b157-34786d2c7541 | 12345678C |  Cancelada | 2025-03-15 |   2025-03-10 |           201
```

4. Eliminamos datos

```
Unset
-- Borrar una reserva con un ID específico
DELETE FROM hotel.reservas
WHERE reserva_id = 8b42d1e4-6150-48bb-b157-34786d2c7541;
```



```
 reserva_id                           | dni       | estado     | fecha_fin  | fecha_inicio | habitacion_id
--------------------------------------+-----------+------------+------------+--------------+---------------
 ad5ee88c-f11f-4962-80e4-8b709a8d3fd2 | 12345678A | Confirmada | 2025-02-05 |   2025-02-01 |           102
 17040e18-d678-486e-873a-04c1e68431b7 | 12345678J | Confirmada | 2025-01-18 |   2025-01-15 |           101
 8b42d1e4-6150-48bb-b157-34786d2c7541 | 12345678C |  Cancelada | 2025-03-15 |   2025-03-10 |           201

(3 rows)
cqlsh> SOURCE '/tmp/DeleteReserva.cql'
cqlsh> SOURCE '/tmp/GetAllTables.cql'

 dni       | nombre | telefono
-----------+--------+--------------
 12345678C | Carlos | +34600123458
 12345678J |   Juan | +34600123456
 12345678A |    Ana | +34600123457

(3 rows)

 habitacion_id | capacidad | precio_por_noche
---------------+-----------+------------------
           201 |         4 |           150.00
           102 |         2 |            75.00
           101 |         1 |            50.00

(3 rows)

 reserva_id                           | dni       | estado     | fecha_fin  | fecha_inicio | habitacion_id
--------------------------------------+-----------+------------+------------+--------------+---------------
 ad5ee88c-f11f-4962-80e4-8b709a8d3fd2 | 12345678A | Confirmada | 2025-02-05 |   2025-02-01 |           102
 17040e18-d678-486e-873a-04c1e68431b7 | 12345678J | Confirmada | 2025-01-18 |   2025-01-15 |           101
```

Conexión al SGBD desde Java

Para conectarnos desde Java debemos incluir una dependencia del conector de Cassandra en el archivo *pom.xml* (para Maven):

```
Unset
<dependency>
    <groupId>com.datastax.oss</groupId>
    <artifactId>java-driver-core</artifactId>
    <version>4.15.0</version>
</dependency>
```

A continuación, ya podemos conectarnos a Cassandra desde una aplicación de Java:

```
Java
import com.datastax.oss.driver.api.core.CqlSession;
import java.net.InetSocketAddress;

public class CassandraConnection {
    public static void main(String[] args) {
        try (CqlSession session = CqlSession.builder()
                .addContactPoint(new InetSocketAddress("127.0.0.1", 9042))
                .withLocalDatacenter("datacenter1")
```

```
                .build()) {

            System.out.println("Conexión exitosa a Cassandra");

            // Ejemplo de consulta
            session.execute("SELECT release_version FROM system.local");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Cassandra en la gestión del hotel

Cassandra tiene diversas ventajas como la alta disponibilidad gracias a ser un sistema distribuido con replicación automática, lo que garantiza que los datos estén disponibles incluso en caso de fallos en los nodos. Además, si se espera un gran volumen de datos, Cassandra permite escalar fácilmente añadiendo nodos al clúster. También es ideal para sistemas con alta tasa de escrituras (como reservas o registros de actividad) y para diseñar estructuras optimizadas para casos de uso específicos, como búsquedas rápidas de reservas por cliente o fechas.

Sin embargo, Cassandra no admite claves foráneas, un elemento importante para asegurar integridad entre entidades relacionales (como reservas y clientes). Cassandra también puede dificultar las consultas que implican múltiples entidades relacionadas por la falta de joins.

Podemos concluir con que Casandra no sería demasiado útil en un sistema de gestión de un hotel, donde las relaciones entre entidades son críticas. Un SGBD relacional como MySQL podría ser más adecuado por su soporte nativo e integridad referencial y consultas complejas.