**✅ Requisitos previos**

* Docker instalado en tu máquina  
  Puedes verificarlo con: docker --version

**🛠️ Paso 1: Crear el contenedor Cassandra**

Ejecuta el siguiente comando para levantar Cassandra con Docker:

docker run --name cassandra-container -d -p 9042:9042 cassandra:latest

🔹 Explicación:

* --name cassandra-container: nombre del contenedor
* -d: modo "detached" (en segundo plano)
* -p 9042:9042: expone el puerto CQL para que puedas conectarte
* cassandra:latest: imagen oficial de Cassandra desde Docker Hub

**🕐 Paso 2: Esperar a que arranque Cassandra**

Cassandra puede tardar un poco (30–60 segundos). Puedes monitorear los logs así:

docker logs -f cassandra-container

Cuando veas algo como Startup complete, ya está listo para usarse.

**🧪 Paso 3: Conectarse al contenedor**

Puedes entrar al contenedor con:

docker exec -it cassandra-container cqlsh

Esto abrirá la CLI de Cassandra (cqlsh) donde puedes ejecutar comandos CQL como:

CREATE KEYSPACE test WITH replication = {'class': 'SimpleStrategy', 'replication\_factor': 1};

**🧼 (Opcional) Detener y eliminar el contenedor**

docker stop cassandra-container docker rm cassandra-container

¿Quieres una versión usando Docker Compose o con volumen persistente?

**You said:**

dame un codigo java que interactue con al anterio contenedor de cassandra

**ChatGPT said:**

Claro, aquí tienes un **ejemplo mínimo en Java** que se conecta al contenedor de Cassandra que creaste con Docker y ejecuta operaciones básicas (crear keyspace, tabla, insertar y leer datos).

**☕ Java + Cassandra - Ejemplo básico**

**📦 1. Dependencias (Maven)**

Agrega esto a tu pom.xml:

<dependencies> <dependency> <groupId>com.datastax.oss</groupId> <artifactId>java-driver-core</artifactId> <version>4.17.0</version> *<!-- o última versión estable -->* </dependency> </dependencies>

⚠️ Usa 4.x ya que es compatible con Cassandra 3.x y 4.x.

**🧩 2. Código Java de ejemplo**

import com.datastax.oss.driver.api.core.CqlSession; import com.datastax.oss.driver.api.core.cql.\*; import java.net.InetSocketAddress; public class CassandraExample { public static void main(String[] args) { try (CqlSession session = CqlSession.builder() .addContactPoint(new InetSocketAddress("localhost", 9042)) .withLocalDatacenter("datacenter1") *// Cassandra usa "datacenter1" por defecto* .build()) { *// Crear Keyspace* session.execute("CREATE KEYSPACE IF NOT EXISTS test WITH replication = " + "{'class': 'SimpleStrategy', 'replication\_factor': 1}"); *// Usar Keyspace* session.execute("USE test"); *// Crear tabla* session.execute("CREATE TABLE IF NOT EXISTS users (id UUID PRIMARY KEY, name text)"); *// Insertar datos* session.execute("INSERT INTO users (id, name) VALUES (uuid(), 'Alice')"); *// Leer datos* ResultSet rs = session.execute("SELECT \* FROM users"); *// Imprimir resultados* for (Row row : rs) { System.out.println("User: " + row.getUuid("id") + " - " + row.getString("name")); } } catch (Exception e) { e.printStackTrace(); } } }

**🔧 3. Compilar y ejecutar**

Compílalo y ejecútalo con Maven o tu IDE favorito (IntelliJ, Eclipse).

**🧪 Verifica conexión**

* Asegúrate que el contenedor Cassandra está corriendo:

docker ps

* Confirma que está escuchando en el puerto 9042 y que el datacenter es datacenter1.

Puedes verificar el datacenter con:

docker exec -it cassandra-container nodetool status

¿Te gustaría que te dé este mismo ejemplo usando **Spring Boot** o **Docker Compose** también?

**You said:**

/Users/josesaidolanogarcia/Library/Java/JavaVirtualMachines/openjdk-23.0.2/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea\_rt.jar=50483 -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath /Users/josesaidolanogarcia/REPOSITORIOS/Cometa/Lesson17\_Cassandra/Java-Cassandra/target/classes:/Users/josesaidolanogarcia/.m2/repository/com/datastax/cassandra/cassandra-driver-core/3.1.2/cassandra-driver-core-3.1.2.jar:/Users/josesaidolanogarcia/.m2/repository/io/netty/netty-handler/4.0.37.Final/netty-handler-4.0.37.Final.jar:/Users/josesaidolanogarcia/.m2/repository/io/netty/netty-codec/4.0.37.Final/netty-codec-4.0.37.Final.jar:/Users/josesaidolanogarcia/.m2/repository/com/google/guava/guava/16.0.1/guava-16.0.1.jar:/Users/josesaidolanogarcia/.m2/repository/io/dropwizard/metrics/metrics-core/3.1.2/metrics-core-3.1.2.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/jnr/jnr-ffi/2.0.7/jnr-ffi-2.0.7.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/jnr/jffi/1.2.10/jffi-1.2.10.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/jnr/jffi/1.2.10/jffi-1.2.10-native.jar:/Users/josesaidolanogarcia/.m2/repository/org/ow2/asm/asm/5.0.3/asm-5.0.3.jar:/Users/josesaidolanogarcia/.m2/repository/org/ow2/asm/asm-commons/5.0.3/asm-commons-5.0.3.jar:/Users/josesaidolanogarcia/.m2/repository/org/ow2/asm/asm-analysis/5.0.3/asm-analysis-5.0.3.jar:/Users/josesaidolanogarcia/.m2/repository/org/ow2/asm/asm-tree/5.0.3/asm-tree-5.0.3.jar:/Users/josesaidolanogarcia/.m2/repository/org/ow2/asm/asm-util/5.0.3/asm-util-5.0.3.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/jnr/jnr-x86asm/1.0.2/jnr-x86asm-1.0.2.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/jnr/jnr-posix/3.0.27/jnr-posix-3.0.27.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/jnr/jnr-constants/0.9.0/jnr-constants-0.9.0.jar:/Users/josesaidolanogarcia/.m2/repository/com/datastax/oss/java-driver-core/4.1.0/java-driver-core-4.1.0.jar:/Users/josesaidolanogarcia/.m2/repository/com/datastax/oss/native-protocol/1.4.5/native-protocol-1.4.5.jar:/Users/josesaidolanogarcia/.m2/repository/com/datastax/oss/java-driver-shaded-guava/25.1-jre/java-driver-shaded-guava-25.1-jre.jar:/Users/josesaidolanogarcia/.m2/repository/com/typesafe/config/1.3.3/config-1.3.3.jar:/Users/josesaidolanogarcia/.m2/repository/org/slf4j/slf4j-api/1.7.26/slf4j-api-1.7.26.jar:/Users/josesaidolanogarcia/.m2/repository/org/hdrhistogram/HdrHistogram/2.1.11/HdrHistogram-2.1.11.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/stephenc/jcip/jcip-annotations/1.0-1/jcip-annotations-1.0-1.jar:/Users/josesaidolanogarcia/.m2/repository/com/github/spotbugs/spotbugs-annotations/3.1.12/spotbugs-annotations-3.1.12.jar:/Users/josesaidolanogarcia/.m2/repository/com/google/code/findbugs/jsr305/3.0.2/jsr305-3.0.2.jar:/Users/josesaidolanogarcia/.m2/repository/com/datastax/oss/java-driver-query-builder/4.1.0/java-driver-query-builder-4.1.0.jar:/Users/josesaidolanogarcia/.m2/repository/io/netty/netty-transport/4.1.71.Final/netty-transport-4.1.71.Final.jar:/Users/josesaidolanogarcia/.m2/repository/io/netty/netty-common/4.1.71.Final/netty-common-4.1.71.Final.jar:/Users/josesaidolanogarcia/.m2/repository/io/netty/netty-buffer/4.1.71.Final/netty-buffer-4.1.71.Final.jar:/Users/josesaidolanogarcia/.m2/repository/io/netty/netty-resolver/4.1.71.Final/netty-resolver-4.1.71.Final.jar:/Users/josesaidolanogarcia/.m2/repository/org/slf4j/slf4j-simple/1.7.36/slf4j-simple-1.7.36.jar com.mx.josesaid.CassandraExample [main] INFO com.datastax.oss.driver.internal.core.DefaultMavenCoordinates - DataStax Java driver for Apache Cassandra(R) (com.datastax.oss:java-driver-core) version 4.1.0 [s0-admin-0] INFO com.datastax.oss.driver.internal.core.time.Clock - Could not access native clock (see debug logs for details), falling back to Java system clock [s0-admin-1] WARN com.datastax.oss.driver.internal.core.control.ControlConnection - [s0] Error connecting to localhost/127.0.0.1:9042, trying next node (ConnectionInitException: [s0|control|connecting...] init query STARTUP: error writing ) com.datastax.oss.driver.api.core.AllNodesFailedException: Could not reach any contact point, make sure you've provided valid addresses (showing first 1, use getErrors() for more: localhost/127.0.0.1:9042: com.datastax.oss.driver.api.core.connection.ConnectionInitException: [s0|control|connecting...] init query STARTUP: error writing ) at com.datastax.oss.driver.api.core.AllNodesFailedException.copy(AllNodesFailedException.java:95) at com.datastax.oss.driver.internal.core.util.concurrent.CompletableFutures.getUninterruptibly(CompletableFutures.java:148) at com.datastax.oss.driver.api.core.session.SessionBuilder.build(SessionBuilder.java:334) at com.mx.josesaid.CassandraExample.main(CassandraExample.java:17)

**ChatGPT said:**

Gracias por compartir el log completo. El error clave es este:

AllNodesFailedException: Could not reach any contact point ...

localhost/127.0.0.1:9042: ... init query STARTUP: error writing

**🧭 Diagnóstico del problema**

El driver **no puede conectarse a Cassandra** en localhost:9042.

Esto puede deberse a varias razones. Vamos paso por paso.

**✅ Verifica estas cosas:**

**1. ¿El contenedor Cassandra está corriendo?**

Ejecuta:

docker ps

Deberías ver algo como:

CONTAINER ID IMAGE ... PORTS

xxxxxx cassandra ... 0.0.0.0:9042->9042/tcp

Si **no está corriendo**, reinícialo:

docker start cassandra-container

**2. ¿Cassandra está listo?**

El contenedor puede estar activo pero Cassandra no terminado de iniciar.

Haz un seguimiento de los logs:

docker logs -f cassandra-container

Busca una línea como:

Starting listening for CQL clients on /0.0.0.0:9042

y

Startup complete

Si no ves eso, espera o reinicia el contenedor.

**3. ¿Tu Java App se conecta a**datacenter1**?**

El driver necesita saber el **nombre correcto del datacenter**.

En tu código Java tienes:

.withLocalDatacenter("datacenter1")

Verifica que ese sea el correcto con:

docker exec -it cassandra-container nodetool status

Resultado esperado (por ejemplo):

Datacenter: datacenter1

Si es otro nombre (ej: "dc1"), cambia el código.

**4. ¿El puerto 9042 está abierto?**

Desde terminal, prueba:

telnet localhost 9042

Si no se conecta, puede que:

* El contenedor no esté exponiendo el puerto
* Estés corriendo Cassandra en una red diferente

Asegúrate de haberlo iniciado con:

docker run --name cassandra-container -d -p 9042:9042 cassandra:latest

**5. (Opcional) ¿Estás usando Apple Silicon (M1/M2)?**

En Macs con chip M1/M2, Cassandra en contenedor puede dar problemas por arquitectura. Puedes usar una imagen específica de Cassandra que funciona con ARM:

docker run --name cassandra-container-arm -d -p 9042:9042 arm64v8/cassandra:latest

**✔️ Alternativas para debugging rápido**

1. **Conéctate desde dentro del contenedor**:

docker exec -it cassandra-container cqlsh

Si funciona ahí, Cassandra está bien, y el problema es de red desde tu host.

**🛠️ ¿Qué hacer ahora?**

Haz estas 3 cosas en orden:

1. Asegúrate de que Cassandra arrancó y dice Startup complete
2. Verifica el datacenter con nodetool status
3. Prueba conectarte con:

.withLocalDatacenter("datacenter1") .addContactPoint(new InetSocketAddress("127.0.0.1", 9042))