Pregunta:

Hola Gemini, He creado un contenedor docker con una versión de antigua de mysql, la 5.0, mediante docker-compose la he conectado a un servidor nest, pero me lanza el error de que la IP del host del servidor nest no está autorizado a conectarse. Tengo bien escritos los datos de conexión a la base de datos mysql, ya que lo he probado conectando a esta base de datos desde fuera de docker y funciona. ¿Qué puede estar ocurriendo para que me lance ese error? ¿Quizás mysql sólo permite la conexión de ciertas IPs?

Respuesta:

Hola. Es un problema común cuando se trabaja con MySQL y Docker. Aquí te detallo las causas más probables y cómo solucionarlas:

### Posibles causas y soluciones

1. MySQL solo escucha en localhost dentro del contenedor:  
   Por defecto, algunas configuraciones de MySQL pueden hacer que el servidor solo escuche conexiones provenientes de localhost (127.0.0.1) dentro del propio contenedor. Cuando tu servidor NestJS intenta conectarse, lo hace desde una IP diferente (la IP interna que Docker asigna al contenedor de NestJS en la red de Docker Compose).
   * **Solución:** Debes configurar MySQL para que escuche en todas las interfaces disponibles (0.0.0.0). Esto se hace modificando el archivo de configuración my.cnf (o my.ini en Windows) de MySQL.  
     Dentro del contenedor Docker, la ruta común para este archivo es /etc/mysql/my.cnf o /etc/my.cnf. Busca la línea que dice:  
     bind-address = 127.0.0.1  
       
     Y cámbiala a:  
     bind-address = 0.0.0.0  
       
     Si no encuentras bind-address, es posible que no esté explícitamente configurado y MySQL ya esté escuchando en 0.0.0.0. En ese caso, la causa sería otra.  
     ¿Cómo modificar my.cnf en Docker?  
     La mejor manera es crear un archivo my.cnf personalizado en tu máquina local y luego montarlo como un volumen en el contenedor de MySQL mediante tu docker-compose.yml.  
     **Ejemplo de my.cnf (en tu máquina local, junto a tu docker-compose.yml):**  
     [mysqld]  
     bind-address = 0.0.0.0  
       
     **Ejemplo de docker-compose.yml:**  
     YAML  
     version: '3.8'  
     services:  
      mysql:  
      image: mysql:5.0 # Asegúrate de usar la versión correcta de MySQL 5.0 si existe una etiqueta específica  
      container\_name: mi\_mysql\_50  
      environment:  
      MYSQL\_ROOT\_PASSWORD: tu\_password\_root  
      MYSQL\_DATABASE: tu\_base\_de\_datos  
      MYSQL\_USER: tu\_usuario  
      MYSQL\_PASSWORD: tu\_password  
      ports:  
      - "3306:3306"  
      volumes:  
      - ./my.cnf:/etc/mysql/my.cnf # Monta tu archivo my.cnf personalizado  
      # Otros volúmenes de datos si los necesitas  
       
      nest\_app:  
      build:  
      context: .  
      dockerfile: Dockerfile  
      container\_name: mi\_nest\_app  
      ports:  
      - "3000:3000"  
      environment:  
      DATABASE\_HOST: mysql # Usa el nombre del servicio de MySQL como host  
      DATABASE\_PORT: 3306  
      DATABASE\_USER: tu\_usuario  
      DATABASE\_PASSWORD: tu\_password  
      DATABASE\_NAME: tu\_base\_de\_datos  
      depends\_on:  
      - mysql
2. Usuario de MySQL sin los permisos adecuados para conectarse desde cualquier host (%):  
   Aunque hayas configurado el usuario de MySQL para conectarse desde fuera de Docker, es crucial que ese usuario tenga permisos para conectarse desde cualquier host (%) y no solo desde localhost.
   * **Solución:** Conéctate a tu base de datos MySQL (puedes hacerlo desde un cliente MySQL en tu máquina local, si está expuesto el puerto) y ejecuta el siguiente comando SQL para tu usuario:  
     SQL  
     -- Si tu usuario es 'tu\_usuario' y tu contraseña 'tu\_password'  
     GRANT ALL PRIVILEGES ON tu\_base\_de\_datos.\* TO 'tu\_usuario'@'%' IDENTIFIED BY 'tu\_password';  
     FLUSH PRIVILEGES;  
       
     O si ya tienes el usuario, simplemente modifica el host:  
     SQL  
     ALTER USER 'tu\_usuario'@'localhost' IDENTIFIED BY 'tu\_password'; -- Si ya existe con localhost  
     -- Luego, si quieres que pueda conectarse desde cualquier IP  
     CREATE USER 'tu\_usuario'@'%' IDENTIFIED BY 'tu\_password';  
     GRANT ALL PRIVILEGES ON tu\_base\_de\_datos.\* TO 'tu\_usuario'@'%';  
     FLUSH PRIVILEGES;  
       
     **Importante:** La sintaxis exacta para GRANT puede variar ligeramente en MySQL 5.0, pero el concepto es el mismo: asegurarte de que el usuario tiene permisos para conectarse desde '%' (cualquier host).
3. Problemas con el nombre del host en la configuración de NestJS:  
   Dentro de una red Docker Compose, los contenedores pueden comunicarse entre sí utilizando los nombres de los servicios definidos en el docker-compose.yml. Si en tu configuración de NestJS estás usando localhost o la IP de tu máquina host, eso es incorrecto.
   * **Solución:** En tu configuración de NestJS, para el host de la base de datos, debes usar el nombre del servicio de MySQL tal como lo definiste en tu docker-compose.yml. En el ejemplo anterior, sería mysql.

TypeScript  
// En tu configuración de NestJS (por ejemplo, en un archivo .env o en el módulo de la base de datos)  
// database.config.ts o .env  
HOST\_MYSQL: 'mysql', // <-- Aquí debe ir el nombre del servicio de MySQL en docker-compose.yml  
PORT\_MYSQL: 3306,  
USER\_MYSQL: 'tu\_usuario',  
PASSWORD\_MYSQL: 'tu\_password',  
DATABASE\_MYSQL: 'tu\_base\_de\_datos',

1. Firewall del contenedor MySQL (menos probable pero posible):  
   Aunque raro en configuraciones Docker por defecto, podría haber alguna regla de firewall dentro del contenedor de MySQL que esté bloqueando las conexiones. Esto es muy poco probable si estás usando una imagen oficial de MySQL.

### Pasos para depurar:

1. Verifica los logs del contenedor MySQL:  
   Ejecuta docker logs mi\_mysql\_50 (reemplaza mi\_mysql\_50 con el nombre de tu contenedor MySQL) para ver si hay mensajes de error específicos de MySQL que indiquen por qué se rechaza la conexión.
2. Intenta conectarte a MySQL desde el contenedor NestJS:  
   Puedes entrar al contenedor de NestJS y tratar de conectarte a MySQL desde allí para confirmar si el problema es la conectividad de red o la autenticación de MySQL.  
   Bash  
   docker exec -it mi\_nest\_app bash # Entra al contenedor de NestJS  
   # Una vez dentro, si tienes el cliente mysql-client instalado  
   mysql -h mysql -u tu\_usuario -p tu\_base\_de\_datos  
     
   Si no tienes el cliente MySQL instalado en el contenedor de NestJS, puedes instalarlo temporalmente o usar herramientas como ping mysql para verificar la resolución de nombres.
3. Asegúrate de que la versión 5.0 de MySQL es compatible con el driver de NestJS:  
   MySQL 5.0 es bastante antiguo. Asegúrate de que el driver MySQL/ORM que estás utilizando en NestJS (por ejemplo, TypeORM con mysql2 o mysql) sea compatible con una versión tan antigua de MySQL. Si el driver espera características más modernas, podría haber problemas de compatibilidad que se manifiesten como errores de conexión.

Al abordar estas posibles causas, lo más probable es que encuentres la solución a tu problema de conexión. ¡Mucha suerte!