Troubleshooting Report: Spring Boot & Oracle Database Connection

This document outlines the step-by-step troubleshooting process for connecting the Graduation Project Spring Boot application to an Oracle 21c database on an Ubuntu server.

Summary of Issues

The primary issue was a **critical lack of server resources (RAM)**, which prevented the Oracle database from starting correctly. This core problem then caused a cascade of secondary errors, including service startup failures, listener problems, and connection timeouts.

The troubleshooting process was divided into three phases:

1. **Server Resource Failure:** Identifying and fixing the server's lack of RAM.
2. **Oracle Service & Startup Errors:** Manually starting the database when automated scripts failed.
3. **Spring Boot Configuration Errors:** Correcting the application's connection string and credentials.

Part 1: Server Resource Failure (The Root Cause)

This was the most critical problem. The server (with only 2GB of RAM and 0 Swap) was not powerful enough to run both the Oracle database and the Java application.

Symptoms:

- **ORA-00845: MEMORY_TARGET not supported on this system**
  - **Diagnosis:** Oracle's Automatic Memory Management (MEMORY_TARGET) requires a large block of shared memory (/dev/shm). The server's default size was too small.
  - **Solution:** We temporarily increased the size of /dev/shm to 4GB.
  - **Command:** sudo mount -t tmpfs shmfs -o size=4G /dev/shm

- **HikariPool-1 - Thread starvation or clock leap detected**
  - **Diagnosis:** The application hung for over 20 minutes. top and free -h commands revealed the server had **0 available RAM** and a load average of 72. The system was "starved" of resources and grinding to a halt.
  - **Solution:** We created and activated a 4GB swap file to act as emergency "slow RAM."
  - **Commands:**
    ```
    sudo fallocate -l 4G /swapfile
    sudo chmod 600 /swapfile
    sudo mkswap /swapfile
    sudo swapon /swapfile
    ```

Part 2: Oracle Service & Startup Errors

Because the server was resource-starved, the automated Oracle scripts failed, forcing us to start everything manually.

Symptoms:

- **lsnrctl status showed The listener supports no services**
  - **Diagnosis:** The listener (the database's "receptionist") was running, but the database itself was offline and had not registered with it.
  - **Solution:** We needed to manually start the database.

- **systemctl start oracle-xe-21c.service Failed**
  - **Diagnosis:** The official service script was unreliable, likely due to the memory issues and conflicts (e.g., trying to start a listener that was already running).
  - **Solution:** We abandoned the service script and proceeded with a full manual startup.

- **SP2-0750: You may need to set ORACLE_HOME & ORA-12162: TNS:net service name is incorrectly specified**
  - **Diagnosis:** When trying to log in as the oracle user, the shell session was missing the environment variables that tell sqlplus where the database software is and what its name is.
  - **Solution:** We manually set the variables before running any commands.
  - **Commands:**

    ```
    sudo su - oracle
    export ORACLE_HOME=/opt/oracle/product/21c/dbhomeXE
    export ORACLE_SID=XE
    export PATH=$ORACLE_HOME/bin:$PATH
    ```

- **ORA-12541: Cannot connect. No listener**
  - **Diagnosis:** After rebooting the server, the listener service was not running.
  - **Solution:** We had to manually start the listener *before* starting the database.
  - **Command:** lsnrctl start (run as the oracle user after setting the environment).

Part 3: Spring Boot Configuration Errors

These were configuration errors in the application.properties file that we discovered while trying to connect.

Symptoms:

- **ORA-12514: Service orclpdb is not registered**
  - **Diagnosis:** The spring.datasource.url was pointing to a service name (orclpdb) that didn't exist.
  - **Solution:** We changed the URL to use the correct SID.

- **ORA-12505: SID XE is not registered**
  - **Diagnosis:** This was a misleading error. The SID *was* XE, but the database was offline (see Part 2), so the listener didn't know about it.
  - **Solution:** The URL format was correct, but we had to fix the database itself.
  - **Final URL:** spring.datasource.url=jdbc:oracle:thin:@localhost:1521:XE

- **ORA-01017: invalid username/password; logon denied**
  - **Diagnosis:** The application was successfully connecting, but the spring.datasource.username or spring.datasource.password was incorrect.
  - **Solution:** updated the application.properties file with the correct credentials (e.g., username system and its password).

The Server's Permanent Manual Startup Procedure

After you reboot this server, you will need to follow these steps every time to get the application running:

1. **Fix Server Memory:**
   ```
   sudo mount -t tmpfs shmfs -o size=4G /dev/shm
   sudo swapon /swapfile
   ```

2. **Start Oracle Services Manually:**
   ```
   # 1. Switch to the oracle user
   sudo su - oracle

   # 2. Set the environment
   export ORACLE_HOME=/opt/oracle/product/21c/dbhomeXE
   export ORACLE_SID=XE
   export PATH=$ORACLE_HOME/bin:$PATH
   ```

```
# 3. Start the listener
lsnrctl start

# 4. Log in and start the database
sqlplus / as sysdba
SQL> startup;
SQL> exit

# 5. Return to your user
exit
```

3. **Run Your Spring Boot Application:**

   Bash

   ```
   cd ~/Teeth-Management-System/BackEnd/
   java -jar target/GraduationProject1-0.0.1-SNAPSHOT.jar
   ```