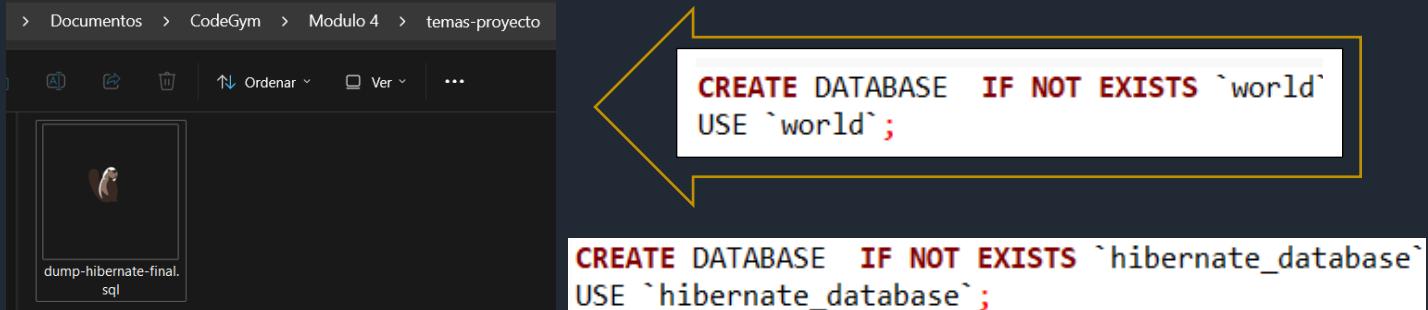


1 Comando de ejecución del servidor MySQL como contenedor docker

```
docker run -d ` 
--name hibernate-db ` 
-e MYSQL_ROOT_PASSWORD=root ` 
-e MYSQL_DATABASE=hibernate_database ` 
-p 3310:3306 ` 
mysql:8.0 ` 
--default-authentication-plugin=mysql_native_password
```

2 Ubicación y modificación de la BD - dump-hibernate-final.sql



3 Importación del archivo dump.sql en la base de datos

```
PS C:\Users\CONRRAD> docker cp "C:\Users\CONRRAD\Documents\CodeGym\Modulo 4\temas-proyecto\dump-hibernate-final.sql" hibernate-db:/dump.sql
Successfully copied 236kB to hibernate-db:/dump.sql
PS C:\Users\CONRRAD> Get-Content "C:\Users\CONRRAD\Documents\CodeGym\Modulo 4\temas-proyecto\dump-hibernate-final.sql" | docker exec -i hibernate-db mysql -u root -proot hibernate_database
mysql: [Warning] Using a password on the command line interface can be insecure.
PS C:\Users\CONRRAD> docker exec -it hibernate-db mysql -u root -proot
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 21
Server version: 8.0.43 MySQL Community Server - GPL

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> Show databases;
+-----+
| Database |
+-----+
| hibernate_database |
| information_schema |
| mysql |
| performance_schema |
| sys |
| world |
+-----+
6 rows in set (0.01 sec)

mysql> USE hibernate_database;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
+-----+
| Tables_in_hibernate_database |
+-----+
| city |
| country |
| country_language |
+-----+
3 rows in set (0.00 sec)
```

- `docker cp "C:\Users\CONRRAD\Documents\CodeGym\Modulo 4\temas-proyecto\dump-hibernate-final.sql" hibernate-db:/dump.sql`
Successfully copied 236kB to hibernate-db:/dump.sql
- `Get-Content "C:\Users\CONRRAD\Documents\CodeGym\Modulo 4\temas-proyecto\dump-hibernate-final.sql" | docker exec -i hibernate-db mysql -u root -proot hibernate_database`
- `docker exec -it hibernate-db mysql -u root -proot`
- `mysql> Show databases;`

Database
hibernate_database
information_schema
mysql
performance_schema
sys
world

- `mysql> USE hibernate_database;`
- `mysql> SHOW TABLES;`

Tables_in_hibernate_database
city
country
country_language

4 DBaver

Docker container MySQL

Name	Container ID	Image	Port(s)	CPU	Actions
hibernate-db	62186d8e0c3c	mysql:8.0	3310:3306	0	

Selección de la BD

Puerto: 3310 Contraseña: root

MySQL ajustes de conexión

General Driver properties SSH SSL

Nombre de usuario: root Contraseña: ****

localhost 2 localhost:3310

Databases

hibernate_database

- Tables
 - city
 - country
 - country_language

4 Conexión a la BD MySQL - Dentro del main

```
private SessionFactory prepareRelationalDb() {
    final SessionFactory sessionFactory;
    Properties properties = new Properties();
    properties.put(Environment.DIALECT, "org.hibernate.dialect.MySQL8Dialect");
    properties.put(Environment.DRIVER, "com.p6spy.engine.spy.P6SpyDriver");
    properties.put(Environment.URL, "jdbc:p6spy:mysql://localhost:3310/hibernate_database");
    properties.put(Environment.USER, "root");
    properties.put(Environment.PASS, "root");
    properties.put(Environment.CURRENT_SESSION_CONTEXT_CLASS, "thread");
    properties.put(Environment.HBM2DDL_AUTO, "validate");
    properties.put(Environment.STATEMENT_BATCH_SIZE, "100");
```

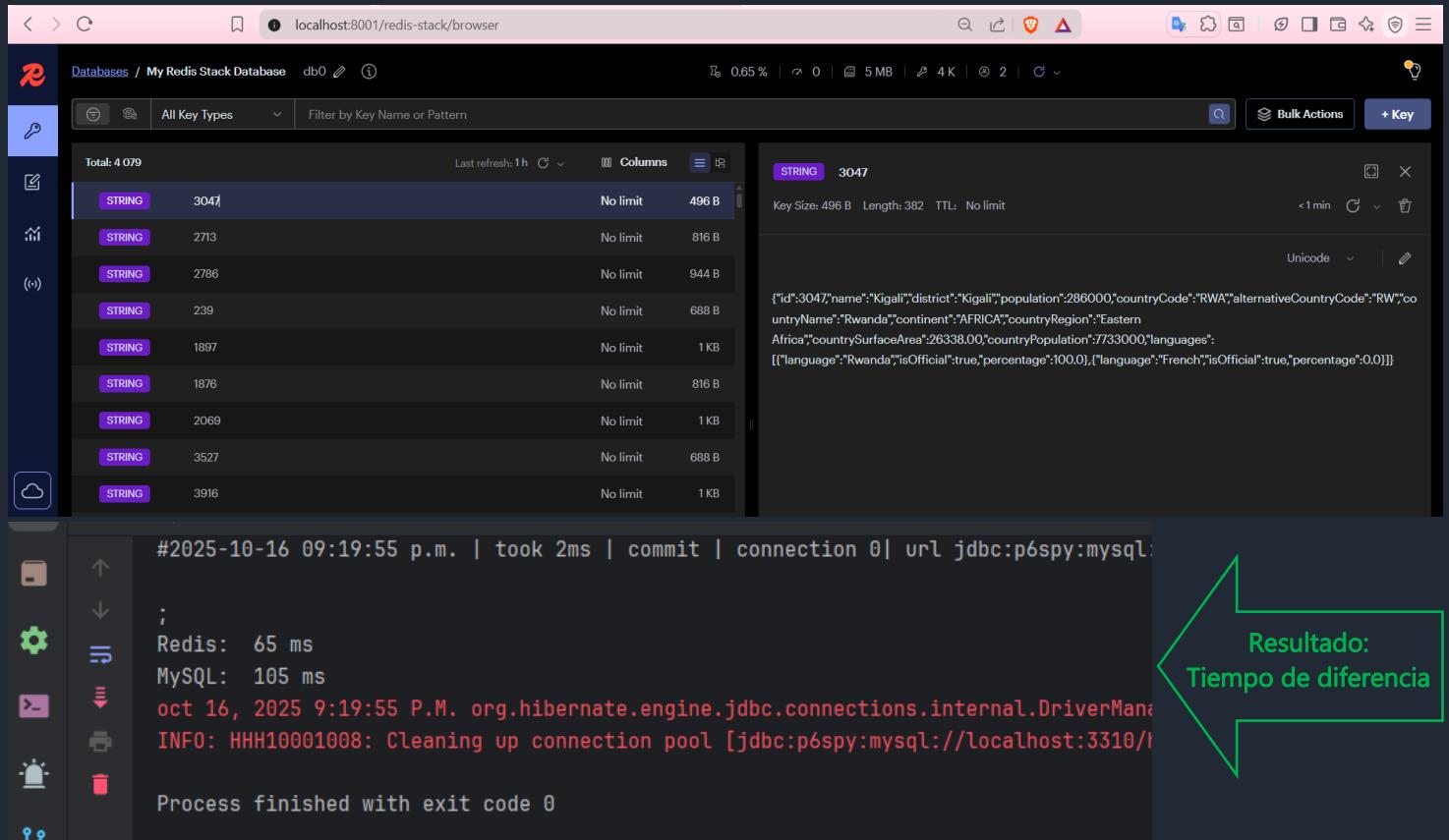
1 Comando de ejecución de Redis como contenedor docker

Argumento	Descripción	Propósito
<code>docker run</code>	Inicia un nuevo contenedor.	El comando base de Docker.
<code>-d</code>	<i>Detached mode.</i>	Ejecuta el contenedor en segundo plano , liberando tu terminal.
<code>--name redis-stack</code>	Asigna un nombre.	Le da un nombre fácil de recordar al contenedor (redis-stack).
<code>-p 6379:6379</code>	Mapeo de Puertos.	Mapea el puerto predeterminado de Redis (6379) del contenedor al puerto 6379 de tu PC.
<code>-p 8001:8001</code>	Mapeo de Puertos.	Mapea el puerto de la Interfaz de Usuario Gráfica (RedisInsight) del contenedor al puerto 8001 de tu PC.
<code>redis/redis-stack:latest</code>	Imagen a Usar.	La imagen específica que contiene el servidor Redis y la interfaz RedisInsight.

```
docker run -d `  
--name redis-stack `  
-p 6379:6379 `  
-p 8001:8001 `  
redis/redis-stack:latest
```

Configuración del cliente Redis - Dentro del main

```
private RedisClient prepareRedisClient() {  
    RedisClient redisClient =  
        RedisClient.create(RedisURI.create("localhost", 6379));  
    try (StatefulRedisConnection<String, String> connection = redisClient.connect()) {  
        System.out.println("\nConnected to Redis");  
    }  
    return redisClient;  
}
```



La captura de pantalla muestra la interfaz web de RedisInsight y una terminal abierta en el fondo.

En la parte superior, la barra de dirección muestra `localhost:8001/redis-stack/browser`. La interfaz de RedisInsight muestra una lista de claves (4079 en total) y un panel detallado para la clave `3047`, que es de tipo STRING con un tamaño de 496 B y una longitud de 382. El valor es un JSON que incluye datos sobre Kigali, Rwanda, así como información sobre idiomas oficiales y no oficiales.

En la terminal abierta debajo, se observan los siguientes logs:

```
#2025-10-16 09:19:55 p.m. | took 2ms | commit | connection 0| url jdbc:p6spy:mysql:  
;  
Redis: 65 ms  
MySQL: 105 ms  
oct 16, 2025 9:19:55 P.M. org.hibernate.engine.jdbc.connections.internal.DriverManager  
INFO: HHH10001008: Cleaning up connection pool [jdbc:p6spy:mysql://localhost:3310/  
Process finished with exit code 0
```

Un cuadro resaltado en verde en la parte inferior derecha indica el resultado:

Resultado: Tiempo de diferencia