Guía Completa: Configuración de Liquibase con Oracle

📋 Resumen del Proyecto

Esta guía documenta el proceso completo para configurar Liquibase con Oracle Database, creando ambientes separados de desarrollo y producción usando Docker.

🎯 Objetivo

- Crear dos bases de datos Oracle separadas (desarrollo y producción)
- Configurar Liquibase para gestionar migraciones de base de datos
- Demostrar el control de versiones de esquemas de base de datos

🛠️ Herramientas Utilizadas

- **Docker**: Para contenedor de Oracle Database
- **Oracle Database XE**: Base de datos
- **Liquibase**: Control de versiones de base de datos
- **SQL Developer**: Cliente gráfico (opcional)

📁 Estructura del Proyecto

```
DATABASESCLASS/
├── project/
│   └── db-migrations/
│       └── changelog/
│           ├── changelog.xml
│           └── changesets/
```

```
│               ├── 001-initial-schema.sql
│               └── 002-add-users-table.sql
├── liquibase-dev.properties
└── liquibase-prod.properties
```

🚀 Pasos Realizados

1. Configuración de Oracle con Docker

Ejecutar contenedor Oracle:

```
docker run -d \
  --name oracle-xe \
  -p 1521:1521 \
  -p 5500:5500 \
  -e ORACLE_PWD=MYPASSWORD \
  gvenzl/oracle-xe:latest
```

Verificar que esté ejecutándose:

```
docker ps
```

2. Creación de Usuarios de Base de Datos

Conectarse como SYS:

```
docker exec -it oracle-xe sqlplus sys/MYPASSWORD!@localhost:1521/XE as
sysdba
```

Crear usuarios para desarrollo y producción:

```
CREATE USER dev_user IDENTIFIED BY dev_password;
GRANT CONNECT, RESOURCE, DBA TO dev_user;
```

```
CREATE USER prod_user IDENTIFIED BY prod_password;
GRANT CONNECT, RESOURCE, DBA TO prod_user;

EXIT;
```

3. Configuración de Archivos Liquibase

liquibase-dev.properties:

```
changeLogFile=project/db-migrations/changelog/changelog.xml
url=jdbc:oracle:thin:@//localhost:1521/XE
username=dev_user
password=dev_password
driver=oracle.jdbc.OracleDriver
```

liquibase-prod.properties:

```
changeLogFile=project/db-migrations/changelog/changelog.xml
url=jdbc:oracle:thin:@//localhost:1521/XE
username=prod_user
password=prod_password
driver=oracle.jdbc.OracleDriver
```

4. Archivo Master Changelog (changelog.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<databaseChangeLog
    xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
    http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.8.xsd">

    <changeSet id="001-initial-schema" author="yourname">
        <sqlFile path="project/db-migrations/changelog/changesets/001-
initial-schema.sql"/>
```

```
    </changeSet>

    <changeSet id="002-add-users-table" author="yourname">
        <sqlFile path="project/db-migrations/changelog/changesets/002-
add-users-table.sql"/>
    </changeSet>

</databaseChangeLog>
```

5. Script de Esquema Inicial (001-initial-schema.sql)

Contenido principal:

- Tabla USERS (usuarios del sistema)
- Tabla TEAMS (equipos de trabajo)
- Tabla EPICS (épicas de proyecto)
- Tabla SPRINTS (sprints de desarrollo)
- Tabla TASKS (tareas)
- Tablas de relación (USER_TEAMS, TASK_ASSIGNMENTS, etc.)
- Tabla ATTACHMENTS (archivos adjuntos)
- Tabla TASK_DEPENDENCIES (dependencias entre tareas)

Nota importante: Se eliminaron índices redundantes para evitar conflictos:

- Oracle crea automáticamente índices para PRIMARY KEY y UNIQUE
- Solo se mantuvieron índices adicionales necesarios

6. Ejecución de Migraciones

Verificar estado antes de aplicar:

```
liquibase --defaultsFile=liquibase-dev.properties status
liquibase --defaultsFile=liquibase-prod.properties status
```

Aplicar migraciones:

```
# Para desarrollo
liquibase --defaultsFile=liquibase-dev.properties update

# Para producción
liquibase --defaultsFile=liquibase-prod.properties update
```

Verificar aplicación exitosa:

```
liquibase --defaultsFile=liquibase-dev.properties status
liquibase --defaultsFile=liquibase-prod.properties status
```

Resultado esperado: "is up to date"

🔧 Resolución de Problemas Encontrados

Problema 1: Error ORA-12541 (No listener)

Causa: Oracle Database no estaba ejecutándose **Solución**: Instalar y ejecutar Oracle via Docker

Problema 2: Error ORA-01408 (Índice duplicado)

Causa: Intentar crear índices que Oracle ya crea automáticamente **Solución**: Remover índices redundantes del script SQL

Problema 3: Error ORA-00955 (Objeto ya existe)

Causa: Tablas ya existían de ejecuciones previas **Solución**: Usar `changelogSync` para marcar changesets como ejecutados

🎯 Conceptos Clave Aprendidos

Separación de Ambientes

- **Desarrollo**: Para pruebas y experimentación
- **Producción**: Para aplicación real con datos reales
- **Aislamiento**: Cambios en DEV no afectan PROD

Control de Versiones de BD

- **Liquibase**: Funciona como Git para bases de datos
- **Changesets**: Unidades atómicas de cambio
- **Rollback**: Capacidad de deshacer cambios si es necesario

Gestión de Esquemas

- **Migraciones**: Aplicar cambios de estructura de forma controlada
- **Sincronización**: Mantener múltiples ambientes coherentes
- **Auditoria**: Historial completo de cambios realizados

📊 Verificación Final

Conexiones SQL Developer:

DEV Connection:

- Host: localhost:1521
- SID: XE
- User: dev_user
- Pass: dev_password

PROD Connection:

- Host: localhost:1521
- SID: XE
- User: prod_user

- Pass: prod_password

Comandos de Verificación:

```
-- Ver tablas creadas
SELECT table_name FROM user_tables;

-- Ver estructura de tabla principal
DESC USERS;

-- Verificar datos (inicialmente vacío)
SELECT COUNT(*) FROM USERS;
```

🏆 Resultados Obtenidos

✅ **Dos bases de datos Oracle separadas funcionando**

✅ **Liquibase configurado y ejecutándose correctamente**

✅ **Estructura de proyecto organizada**

✅ **Migraciones aplicadas exitosamente**

✅ **Control de versiones de esquema implementado**

🔄 Conceptos del Mundo Real

Flujo Típico en Empresas:

1. **Desarrollador** hace cambios en DEV
2. **QA** verifica en ambiente de pruebas
3. **DevOps** aplica a PROD usando Liquibase
4. **Nunca** cambios directos en producción

Beneficios de esta Aproximación:

- **Consistencia**: Misma estructura en todos los ambientes
- **Trazabilidad**: Historial completo de cambios
- **Reversibilidad**: Capacidad de deshacer cambios
- **Colaboración**: Todo el equipo usa los mismos scripts

🎯 Próximos Pasos Sugeridos

1. **Agregar datos de prueba**: Crear changeset con INSERT statements
2. **Implementar rollback**: Crear scripts de reversión
3. **Automatización**: Integrar con CI/CD pipelines
4. **Monitoreo**: Configurar alertas para cambios en PROD

Documento generado el 28 de Mayo, 2025

Proyecto: DatabasesClass - Liquibase Configuration