Contents

1 LIQUIBASE BASICS: WHY ? HOW?

For the SM revamp application we are targeting to build DB-agnostic application. Here what this means is by changing few properties in application we can make it work with various databases. All the SQL needed at application runtime i.e. DML select, insert, update and delete are generated by Hibernate. In this way we as developers only deal with the objects in Java, and need not bother about SQL semantic differences between Databases.

But now challenge is how to make the DB full version in such a way that it becomes DB-agnostic as well. There are below approaches

1. Simplest is to make 1 full version per DB we intend to support. And similar fashion, each hotfix release for each DB as well and Crestel installer should have config of choosing the correct files based on deployed DB in that environment.
   Pros: easy to understand for any new developer and DBA can inspect the scripts, in case things go wrong. Fits with Elitecore experience so far, easy to integrate in current build and deploy practices.

   Cons: Duplicate effort for each DB supported. Developer/Resource needs knowledge of each supported DB.

2. Use hibernate to generate the create scripts for Full version, use in update mode for hotfix and validate for regular operation.
   Pros: less effort, scripts auto generated, need not to deal with DB specific SQL at any time.

   Cons: 1.there are limitations where update mode wont rename columns, change data type, change length etc, detailed n attached document. To handle these need to add script support along with the update approach.

   a. Will be difficult to manage DB version info.
3. Use a third party tool like Liquibase to manage Database versioning.
   Pros: DB agnostic XML way of specifying needed data structures

   Cons: need to learn new way and not sure of stability.

   http://www.liquibase.org/quickstart.html

Relevant Links :

http://dba-presents.com/index.php/other/my-thoughts/34-database-agnostic-applications

As a final decision, Liquibase has been adopted and this document records how it is to be used.

2 STEPS FOR GENERATING CHANGELOG (SAMPLES)

Initially the changelog was generated using the generateChangeLog Ant script. There are 2 ways of doing the same,

- Using *Hibernate as the sample database* to generate changelog
- Using existing DB created through create mode of Hibernate for reference DB to generate changelog.

Both were used and master set was arrived at. This can be run from command line or Ant task config :

```
<target name="genchgLog" description="generates changelogs using Liquibase">
<liquibase:generateChangeLog classpathref="lib.path"    logLevel="debug">

<liquibase:database url="{spring_hibernateUrl}"
user="${oracle_username}" password="${oracle_password}" />

<liquibase:xml outputfile="${genchglog_outputFile}"encoding="UTF-8" />

</liquibase:generateChangeLog>

</target>
```

Going further once the full version DB is created, for any DB change developer has to do following:

1. Incremental changes need to be tracked through diff mechanism. So the developer has to do necessary changes in the application, add classes, change hibernate annotations in existing classes etc.
2. Generate diff of his existing Database which has been created through Liquibase and hibernate configuration.
3. For this purpose the reference database is taken as hibernate:classic:hibernate.cfg.xml. this has all current entities added to it. In future if any new entity class is created, it should be added here.
4. The Liquibase Diff is to be carried out with commandline tool, settings of which are as under:
5. In eclipse create a new Run configuration with mail class as liquibase.integration.commandline.Main And command line arguments like :

```
--logLevel=debug
--url=hibernate:classic:hibernate.cfg.xml
--referenceDriver=org.postgresql.Driver
--referenceUrl=jdbc:postgresql://127.0.0.1:5432/smliquipostgre
--referenceUsername=postgres
```

--referencePassword=vandu83TH

--changeLogFile=chkDiff.xml

--classpath=modules\servermanager\war\WEB-INF\lib\liquibase-hibernate4.2-3.5.jar

diffChangeLog

3 GUIDELINES OF CHANGESET:

1. Each changeset should have a single change (create table, add column, add foreign key constraint etc).
2. Change Set Ids, we will follow the syntax of SM0000000001 and author as Elitecore, this is so that if a change set if to be executed twice it can be detected duplicate. We will use "SM0XXXXXXXXX" id pattern for DDL change logs and "SM2XXXXXXXXX" id pattern form DML change logs.
3. ID length should not be more than 12 characters.
4. Use <comments> in the change sets. They say “A stitch in time saves nine!”
5. Change sets are to be generated using the diff mechanism explained above, but manual verification of the above is needed and these are the possible changes that need to be made.
6. Change VARCHAR2 to VARCHAR.
7. If generated changelog is like VARCHAR (250 CHAR) remove the CHAR inside brackets.
8. If there’s BLOB column in the table, need to write 2 changelogs, one for dbms=postgresql where column type is to be kept as oid and other for MySQL and oracle and data type is kept as LONGBLOB.
9. For int datatype in Java generated Liquibase data type should be INT and for long it should be BIGINT.
10. Any table using Table generation strategy if minimal insert is given, the TBLTPRIMARYKEY table should be updated using custom sql changeset.
11. Sample changelogs for common scenarios with example
    - 1 a. adding a not null column without pre-existing data in table works ex. 1ab_diffChgLog_addColType.xml
    - b. adding a not null column with pre-existing data in table: works ex. 1ab_diffChgLog_addColType.xml
    - 1 c. changing value of nullability from true to false or vice-versa: works 1c_diffChgLog_chgNullability.xml
    - 1d. change column name: chglog is not auto-generated but if coded as per guideline it works ex. 1d_colRename_diffChgLog
    - 2. Removing a column: works drop column is generated
    - 3 a. altering data type of column: not managed This is not auto-detected in diff, BUT CAN BE MANUALLY MANAGED, 3aModify_dataType_diffChgLog
    - 3 b. changing the size of varchar column : This is not auto-detected in diff, BUT CAN BE MANUALLY MANAGED, 3aModify_dataType_diffChgLog

- 4. adding new table: ideally should work but because of bug in 3.4.2 https://liquibase.jira.com/browse/CORE-2679 does not work.{MANUAL} As a workaround we can keep only the new entities in hibernate.cfg, use it as source for generate change log. This changelog when applied to db generates the correct table.
- 5. change table name of existing entity. Fails again with similar trace as above. If manually renameTable changelog command is written, it works well. 4b_rename_tablediffChgLog
- 6 a. relationship between entities change one to one becomes one to many : not tested but logically no impact as FK is used to maintain the relation.
- 6 b. If many to many: Join table will come into picture, rare scenario not tested yet
- 6 c. but if a relation is deleted or removed: relevant change log gets generated and applied. Ref : 6c_mappingDeleted_diffChgLog
- 6d new relation is added among existing tables: relevant change log gets generated and applied. Ref : 6d_newmappingAdded_diffChgLog

12. While adding any new change log, give id as your initials of your firstname and last name and then id: for DDL changelog,it will start from *0XXXXXXXXX and for DML, it will start from 2XXXXXXXXX.*

Ex:JP2000000001 for DML and JP0000000001for DDL

13. If you need to change the existiong csv file. Just make changes in .csv file and runOnChange=*"true" for that table's changelog.* If you have to update existing regex pattern in TBLMREGEX table,make changes in .csv file and the existing changset for that table must be like :

```
<changeSet author="elitecore" id="SM2000000009" runOnChange="true">

		<loadUpdateData encoding="UTF-8"	file="master_data/export_TBLMREGEX.csv"
tableName="TBLMREGEX"		primaryKey="ID">

		</loadUpdateData>

	</changeSet>
```

3.5 Check in XML & CSV files to svn to avoid sequence id conflicts.

14. When you wish to remove some rows from table, remove them from csv file 1st. Then write changelog to delete them from table:

```
<changeSet author="elitecore" id="JP2000000014">

	<delete tableName="tableName">

	<where>columnName in (value1,value2,..,valuen)</where>

	</delete>
```

```
</changeSet>
```

Example:

```
<changeSet author="elitecore" id="JP2000000014">
     <delete tableName="TBLTSNMPALERT">
        <where>NAME in
('gtpPrimeServiceSyncFrom','ipLoggerParsingServiceSyncFrom','natFlowBinaryCollectionServiceSyncFrom','natFlowCollectionServiceSyncFrom')</where>
    </delete>
</changeSet>
```

15. Everyday make a habit of running Liquibase build after taking SVN update, so that the DB in your local environment is in sync with the checked in Liquibase files.

4 STEPS FOR APPLYING CHANGELOG

1. Install database Oracle XE, MySql or PostGreSQL. Create schema in the same and using those details update the properties file path trunk/modules/servermanager/liquibase_export.properties
2. Some sample databases have been uploaded in SVN location http://crestelsvn:3690/svn/crestelmediation/projects/SM_REVAMP/tools
3. Kindly google to get basic installation steps. Standard installations to be done.
4. Steps for creating schema :

In export properties, you need to make below changes:

```
userdriver=appropriate driver accoring to database

        userurl= url according to db

        userusername =root user for db

        userpassword=password of root user

        Ex:

        for oracle,

        userdriver=oracle.jdbc.driver.OracleDriver

        userurl=jdbc:oracle:thin:@127.0.0.1:1521:xe

        userusername =system

        userpassword=manager

        for mysql,

        userdriver=com.mysql.jdbc.Driver

        userurl=jdbc:mysql://127.0.0.1:3306/mysql

        userusername =root

        userpassword=root10
```

Now, you have to give stage name, And then udate the value of username and password accordingly. Username must be crestelsmUserName and password should be crestelsmpwdUserName.

Ex: If stage name is, rewamp then username will be crestelsmrewamp and password will be crestelsmpwdrewamp.

Ex:

```
  Mysql database configuration.

driver=com.mysql.jdbc.Driver

url=jdbc:mysql://127.0.0.1:3306/crestelsmtst

username =crestelsmj2

password=crestelsmpwdj2


for Posgresql,

driver=org.postgresql.Driver

url=jdbc:postgresql://127.0.0.1:5432/postgres

username =crestelsmj2

password=crestelsmpwdj2


for oracle,

driver=oracle.jdbc.driver.OracleDriver

url=jdbc:oracle:thin:@127.0.0.1:1521:xe

username=crestelsmj2

password=crestelsmpwdj2
```

5. Once the properties are updated, fire the ANT build using liquibase_build_export.xml. This script will Schema and user will be created by Liquibase and it will read all the change log files and create the DB to be used for SM revamp.
6. Now update this config in jdbc.properties and start tomcat.

5 ISSUES FACED AND SOLUTIONS DONE

1. Columns with mixed case cause issue in PostGreSQL, so all column names to be kept in upper case only.
2. Any reserved words in any db if used as column names, cause invalid SQL when that table is accessed.
3. Issue related to data type, VARCHAR and BLOB.
4. When changing the value for stage name and username and password and execute the liquibase again, it will throw exception as the crete user command would have been fired already and it's entry will be in db in databasechangelog and the changeset will not match as the values will be altered. So,at that time need to revert DBFV.
5. To revert Database Full Version,

drop user username cascade; //drop user
drop tablespace tblspaceName INCLUDING CONTENTS AND DATAFILES;

//drop table space

drop profile profileName; // drop profile
drop role RoleName; //drop role
delete from databasechangelog; //drop entry from database changelog table

Ex:

drop user crestelsmt1 cascade;
drop tablespace crestelsmt1 INCLUDING CONTENTS AND DATAFILES;
drop profile crestelprofile;
drop role crestelrole;
delete from databasechangelog;

6 BUILD DETAIL

TBD. Automation of creation of DB full build and MR process details.

7 DROP COMMANDS

In case while testing for DB FV we want to drop the created entities, following is order list of commands for Oracle.

drop user crestelsmt1 cascade;

drop tablespace crestelsmt1 INCLUDING CONTENTS AND DATAFILES;

drop profile crestelprofile;

drop role crestelrole;

delete from databasechangelog;

8 LINKS:

Liquibase related useful tasks :

http://www.liquibase.org/documentation/ant/index.html

http://www.liquibase.org/documentation/generating_changelogs.html

http://www.liquibase.org/download/index.html

http://www.liquibase.org/documentation/command_line.html

http://www.liquibase.org/databases.html

http://www.liquibase.org/documentation/changeset.html

http://dba.stackexchange.com/questions/75470/database-migration-with-liquibase-using-different-dbms

http://www.liquibase.org/documentation/changelog_parameters.html

Sample Project using Spring/Hibernate/Liquibase

https://github.com/liquibase/liquibase-hibernate/wiki

https://github.com/abdulwaheed18/SHLIntegration

http://www.waheedtechblog.in/2012/08/how-to-integrate-liquibase-with.html

http://www.baeldung.com/liquibase-refactor-schema-of-java-app

https://dzone.com/articles/liquibase-and-hibernate

Postgre docs

https://www.postgresql.org/docs/9.4/static/sql-createtablespace.html

http://stackoverflow.com/questions/8676239/how-to-create-a-database-in-mysql-sql-server-and-oracle-from-a-single-ant-build

MySQL docs :

http://www.bluepiccadilly.com/2011/12/creating-mysql-database-and-user-command-line-and-bash-script-automate-process