Steps to upgrade our Vmidc Server DB Schema or do Data Migration

Version 1.0  
April 25, 2014  
By Vmidc Team

# A. Steps to add a new column to an existing table:

## 1. Add the new xyz field to our existing entity Pojo class, e.g.

* @Entity
* @Table(name = "APPLIANCE")
* **public** **class** Appliance **extends** BaseEntity {
* **private** **static** **final** **long** *serialVersionUID* = 1L;
* **@Column(name = "xyz") //🡨---------------- new field/column**
* **private String xyz;**
* @Column(name = "model", unique = **true**, nullable = **false**)
* **private** String model;
* @Column(name = "manager\_type", nullable = **false**)
* @Enumerated(EnumType.*STRING*)
* **private** ManagerType managerType;
* @Column(name = "manager\_software\_version", nullable = **false**)
* **private** String managerSoftwareVersion;
* @OneToMany(mappedBy = "appliance", fetch = FetchType.*LAZY*, cascade = CascadeType.*ALL*)
* **private** Set<ApplianceSoftwareVersion> applianceSoftwareVersions = **new** HashSet<ApplianceSoftwareVersion>();
* …

## 2. Add the new xyz column to the sqlStatements array of the com. com.mcafee.vmidc.broker.util.db.upgrade.Schema class, e.g.

**public** **class** Schema {

**private** **static** **final** Logger *log* = Logger.*getLogger*(Schema.**class**);

**public** **final** String[] sqlStatements = { //🡨 this array is used for fresh-install

"create table APPLIANCE (" +

"id bigint generated by default as identity," +

"xyz varchar(255)," + 🡨------------ new column xyz

"created\_by varchar(255)," +

"created\_timestamp timestamp," +

"deleted\_by varchar(255)," +

"deleted\_timestamp timestamp," +

"marked\_for\_deletion boolean," +

"updated\_by varchar(255)," +

"updated\_timestamp timestamp," +

"version bigint," +

"manager\_software\_version varchar(255) not null," +

"manager\_type varchar(255) not null," +

"model varchar(255) not null," +

"primary key (id)" +

");",

…



## 3. Increment the static variable TARGET\_DB\_VERSION in the com. com.mcafee.vmidc.broker.util.db.upgrade.ReleaseUpgradeMgr, e.g.

**…. public** **class** ReleaseUpgradeMgr {

**public** **static** **final** **int** *INITIAL\_DB\_VERSION* = 0;

/\*

\* TARGET\_DB\_VERSION will be manually changed to the real target db version

\* to which we will upgrade

\*/

**public** **static** **final** **int** ***TARGET\_DB\_VERSION*** = *1 //🡨---- 1 is the new target //version, means that we will upgrade db from version 0 to version 1*

**private** **static** **final** Logger *log* = Logger.*getLogger*(ReleaseUpgradeMgr.**class**);

…

## 4. Add your schema rebuild logics in the “switch” statement as seen in the com. com.mcafee.vmidc.broker.util.db.upgrade.ReleaseUpgradeMgr class, e.g.

**…. public** **class** ReleaseUpgradeMgr {

…

.. try {

SessionFactory sessionFactory = HibernateUtil.getSessionFactory();

session = sessionFactory.getCurrentSession();

// We must open a new transaction before doing anything with the

// DB

tx = session.beginTransaction();

switch (curDbVer) {

case 0: // add your SQL upgrade logics here to upgrade ver 0 to 1  
 upgradeToTargetSample(session); //no break here

For example:

Here is the SQL to add new xyz column to exiting Appliance table:

String rebuildSql = “**alter table APPLIANCE add xyz varchar(255)”;**

Here is the sample code to execute the above sql statement:

**private** **static** **void** upgradeToTargetSample(**final** Session session) {

session.doWork(**new** Work() {

@Override

**public** **void** execute(Connection connection) **throws** SQLException {

Statement statement = **null**;

**try** {

// Implement your JDBC-based SQL here

statement = connection.prepareStatement(rebuildSql);

statement.execute();

} **finally** { /\* close the statement \*/

**if** (statement != **null**) {

statement.close();

}

}

}

});

}

# B. Steps to change the existing column properties of an existing table:

- Same as above. Just look at the same code patterns.

# C. Steps to add a new table:

- Same as above. Just look at the same code patterns.

# D. Steps to change existing column value from val1 to val2 (data migration):

- Same as above. Just look at the same code patterns. Just add sql to modify/insert changed values.