**Derby的导入与导出功能详解**

   最近为了解决大容量CSV文件的读取问题不得不转向Derby的学习，因为客户又提出解析CS文件的新需求，而CSVdrive那个鸟东西实在是接口少的 可怜，最新的版本也是在04年出的，幸好现在捞到Derby这根救命稻草，按我的想法，是先将csv文件批量移植到Derby数据库中，然后利用Derby强大丰富的sql语句功能，然后解析数据，必然事半功倍。 由于本项目针对Derby某一功能的使用，由于进度问题，所以对derby其他方面未做介绍，后文再提，事不宜迟，搞移植先！

* 大块文件的导入

derby 提供了两个导入过程来让用户来进行块文件的导入操作：  
      1. 从一个文件或表中导入数据，使用SYSCS\_UTIL.SYSCS\_IMPORT\_TABLE过程，其过程的定义如下：

SYSCS\_UTIL.SYSCS\_IMPORT\_TABLE (IN schemaName VARCHAR(128),

IN tableName VARCHAR(128), IN fileName VARCHAR(32672),

IN columnDelimiter CHAR(1), IN characterDelimiter CHAR(1),

IN codeset VARCHAR(128), IN replace SMALLINT)

        这个过程没有返回结果集  
     2. 从一个文件导入数据到某一表中的列的子集中，使用SYSCS\_UTIL.SYSCS\_IMPORT\_DATA过程，为了导入数据到一个表的列的子集，你需要给出insertColumns参数 。处理过程定义如下：

**SYSCS\_UTIL.SYSCS\_IMPORT\_DATA (IN schemaName VARCHAR(128),**

**IN tableName VARCHAR(128), IN insertColumns VARCHAR(32672),**

**IN columnIndexes VARCHAR(32672), IN fileName VARCHAR(32672),**

**IN columnDelimiter CHAR(1), IN characterDelimiter CHAR(1),**

**IN codeset VARCHAR(128), IN replace SMALLINT)**

这个过程无返回结果集。

* 参数说明

**schemaName：**表的schema名。若设null值，将使用默认的schema名  
**tableName：**数据库的目的表名。该表不能是系统表或已声明的临时表。且字符串要与数据库表明的大小写匹配。若设为null，则会出错。  
**insertColumns：**目的表中的列名集合。列名要用逗号分割。若设为null，则导入数据到所有列。  
**columnIndexes：** 分隔符文件的列序号集合，以逗号分割（从1开始）。若设为null，则会导入该文件的所有列。  
**fileName：**分隔符文件名。若无路径，则使用 当前工作路径。若设为null，则会出错。  
**columnDelimiter：**分隔符文件所使用的分割符，若设为null，默认值为逗号。  
**characterDelimiter：**字符分割符，用特定的分隔符来标志字符串。若设为null，默认为一个双引号。  
**Codeset：**分隔符文件所使用的字符串输入文件。若设为null值，将使用与JVM一致的字符设置。  
**Replace：**非零值的参数将会使用REPLACE模式进行数据文件的导入。如果使用零值，则会使用INSERT模式。REPLACE模式将会删除数据库中已存在的重复记录，而INSERT模式则无论数据库中是否存在重复记录都将直接插入。若设为null，则会出错。

* 数据导入示例：

先在数据库中创建表tab1：  
CREATE TABLE tab1 (c1 CHAR(30), c2 INT GENERATED ALWAYS AS IDENTITY, c3 REAL,c4 CHAR(1))  
      1.   假设你你想导入myfile1.csv到tab1中，myfile1.csv文件中包含以下数据：  
            Robert,45.2,J  
            Mike,76.9,K  
             Leo,23.4,I  
为了从myfile.csv中导入所有数据到tab1的C1、C3、C4列，可以执行SYSCS\_UTIL.SYSCS\_IMPORT\_DATA过程，如下：

**CALL SYSCS\_UTIL.SYSCS\_IMPORT\_DATA (NULL, 'TAB1', 'C1,C3,C4' , null,'myfile1.csv',null, null,null,0)**

    2.   假设你想导入myfile2.csv到tab1中，myfile2.csv文件中包含以下数据：  
Robert,1,45.2,J

Mike,2,23.4,I

Leo,3,23.4,I

为了导入myfile2.csv中的1、3、4列中的数据到数据库的tab1中的C1、C3、C4列，可以执行SYSCS\_UTIL.SYSCS\_IMPORT\_DATA过程，如下：

**CALL SYSCS\_UTIL.SYSCS\_IMPORT\_DATA (NULL, 'TAB1', 'C1,C3,C4' ,**

**'1,3,4','myfile2.csv',null, null,null,0)**

* 通过JDBC执行导入/导出过程

通过JDBC编程也能实现导入/导出过程，下面的代码段向你展现了使用java编程来实现SYSCS\_UTIL.SYSCS\_EXPORT\_TABLE过程，下例中，要从staff数据表中导出数据到文件staff.csv中，并使用百分号（%）分隔符 **PreparedStatement ps=conn.prepareStatement("CALL  
SYSCS\_UTIL.SYSCS\_EXPORT\_TABLE (?,?,?,?,?,?)");  
ps.setString(1,null);  
ps.setString(2,"STAFF");  
ps.setString(3,"staff.dat");  
ps.setString(4,"%");  
ps.setString(5,null);  
ps.setString(6,null);  
ps.execute();**

--------------------------

**Creating, dropping, and backing up databases**

You create new databases and access existing ones by specifying attributes to the Derby connection URL.

There is no drop database command. To drop a database, delete the database directory with operating system commands. The database must not be booted when you remove a database. You can get a list of booted databases with *getPropertyInfo*.

To back up a database, you can use the online backup utility. For information on this utility, see the te<*Derby Server and Administration Guide*te<.

You can also use *roll-forward recovery* to recover a damaged database. Derby accomplishes roll-forward recovery by using a full backup copy of the database, archived logs, and active logs from the most recent time before a failure. For more information on roll-forward recovery see the te<*Derby Server and Administration Guide*te<.

<http://db.apache.org/derby/docs/dev/devguide/cdevdvlp42173.html>